Congestion Exposure (ConEx) Working Group
Internet-Draft
Intended status: Informational
Expires: April 27, 2015
October 24, 2014

M. Mathis
Google, Inc
B. Briscoe
BT

Congestion Exposure (ConEx) Concepts, Abstract Mechanism and Requirements

draft-ietf-conex-abstract-mech-13

Abstract

This document describes an abstract mechanism by which senders inform the network about the congestion recently encountered by packets in the same flow. Today, network elements at any layer may signal congestion to the receiver by dropping packets or by ECN markings, and the receiver passes this information back to the sender in transport-layer feedback. The mechanism described here enables the sender to also relay this congestion information back into the network in-band at the IP layer, such that the total amount of congestion from all elements on the path is revealed to all IP elements along the path, where it could, for example, be used to provide input to traffic management. This mechanism is called congestion exposure or ConEx. The companion document "ConEx Concepts and Use Cases" provides the entry-point to the set of ConEx documentation.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Overview 3
  2.1. Terminology 6
3. Requirements for the ConEx Abstract Mechanism 7
  3.1. Requirements for ConEx Signals 7
  3.2. Constraints on the Audit Function 8
  3.3. Requirements for non-abstract ConEx specifications 9
4. Encoding Congestion Exposure 11
  4.1. Naive Encoding 11
  4.2. Null Encoding 12
  4.3. ECN Based Encoding 12
  4.4. Independent Bits 13
  4.5. Codepoint Encoding 13
  4.6. Units Implied by an Encoding 14
5. Congestion Exposure Components 15
  5.1. Network Devices (Not modified) 15
  5.2. Modified Senders 15
  5.3. Receivers (Optionally Modified) 16
  5.4. Policy Devices 16
    5.4.1. Congestion Monitoring Devices 16
    5.4.2. Rest-of-Path Congestion Monitoring 17
    5.4.3. Congestion Policers 17
  5.5. Audit 18
6. Support for Incremental Deployment 21
7. IANA Considerations 24
8. Security Considerations 24
9. Acknowledgements 25
10. Comments Solicited 26
11. References 26
  11.1. Normative References 26
  11.2. Informative References 26

1. Introduction

This document describes an abstract mechanism by which, to a first approximation, senders inform the network about the congestion encountered by packets earlier in the same flow. It is not a complete protocol specification, because it is known that designing an encoding (e.g. packet formats, codepoint allocations, etc) is likely to entail compromises that preclude some uses of the protocol. The goal of this document is to provide a framework for developing and testing algorithms to evaluate the benefits of the ConEx protocol and to evaluate the consequences of the compromises in various different encoding designs. This document lays out requirements for concrete protocol specifications.

A companion document [RFC6789] provides the entry point to the set of ConEx documentation. It outlines concepts that are pre-requisites to understanding why ConEx is useful, and it outlines various ways that ConEx might be used.

2. Overview

As typical end-to-end transport protocols continually seek out more network capacity, network elements signal whenever congestion results, and the transports are responsible for controlling this network congestion [RFC5681]. The more a transport tries to use capacity that others want to use, the more congestion signals will be attributable to that transport. Likewise, the more transport sessions sustained by a user and the longer the user sustains them, the more congestion signals will be attributable to that user. The goal of ConEx is to ensure that the resulting congestion signals are sufficiently visible and robust, because they are an ideal metric for networks to use as the basis of traffic management or other related functions.

Networks indicate congestion by three possible signals: packet loss, ECN marking or queueing delay. ECN marking and some packet loss may be the outcome of Active Queue Management (AQM), which the network uses to warn senders to reduce their rates. Packet loss is also the natural consequence of complete exhaustion of a buffer or other network resource. Some experimental transport protocols and TCP variants infer impending congestion from increasing queuing delay. However, delay is too amorphous to use as a congestion metric. In this and other ConEx documents, the term 'congestion signals' is generally used solely for ECN markings and packet losses, because they are unambiguous signals of congestion.

In both cases the congestion signals follow the route indicated in Figure 1. A congested network device sends a signal in the data stream on the forward path to the transport receiver, the receiver passes it back to the sender through transport level feedback, and the sender makes some congestion control adjustment.

This document extends the capabilities of the Internet protocol suite with the addition of a new Congestion Exposure signal. To a first approximation this signal, also shown in Figure 1, relays the congestion information from the transport sender back through the internetwork layer where it is visible to any interested internetwork layer devices along the forward path. This document frames the engineering problem of designing the ConEx signal. The requirements are described in Section 3 and some example encoding are presented in Section 4. Section 5 describes all of the protocol components.

This new signal is expressly designed to support a variety of new policy mechanisms that might be used to instrument, monitor or manage traffic. The policy devices are not shown in Figure 1 but might be placed anywhere along the forward data path (see Section 5.4).

```
,---------.                                               ,---------.
|Transport|                                               |Transport|
| Sender  |   .                                           |Receiver |
|         |  /|___________________________________________|         |
|      ,-<---------------Congestion-Feedback-Signals--<--------.    |
|      |   |/                                             |   |     |
|      |   |\           Transport Layer Feedback Flow     |   |     |
|      |   | \ ___________________________________________|   |     |
|      |   |  \|                                          |   |     |
|      |   |   '         ,-----------.               .    |   |     |
|      |   |_____________|           |_______________|\   |   |     |
|      |   |    IP Layer |           |  Data Flow      \  |   |     |
|      |   |             |(Congested)|                  \ |   |     |
|      |   |             |  Network  |--Congestion-Signals---->-'     |
|      |   |             |  Device   |                   \|         |
|      |   |             |           |                   /|         |
|      '----------->--(new)-IP-Layer-ConEx-Signals-------->|         |
|          |             |           |                  / |         |
|          |_____________|           |_______________  /  |         |
|          |             |           |               |/   |         |
'---------'              '-----------'               '    '---------'
```

Figure 1: The Flow of Congestion and ConEx Signals

Since the policy devices can affect how traffic is treated it is assumed that there is an intrinsic motivation for users, applications or operating systems to understate the congestion that they are causing. Therefore, it is important to be able to audit ConEx signals, and to be able to apply sufficient sanction to discourage cheating of congestion policies. The general approach to auditing is to count signals on the forward path to confirm that there are never fewer ConEx signals than congestion signals. Many ConEx design constraints come from the need to assure that the audit function is sufficiently robust. The audit function is described in Section 5.5, however significant portions of this document (and prior research [Refb-dis]) is motivated by issues relating to the audit function and making it robust.

The congestion and ConEx signals shown in Figure 1 represent a series of discrete events: ECN marks or lost packets, carried by the forward data stream and fed back into the Internetwork layer. The policy and audit functions are most likely to act on the accumulated values of these signals, for which we use the term "volume". For example traffic volume is the total number of bytes delivered, optionally over a specified time interval and over some aggregate of traffic (e.g. all traffic from a site). While loss-volume is the total amount of bytes discarded from some aggregate over an interval. The term congestion-volume is defined precisely in [RFC6789]. Note that volume per unit time is (average) rate.

A design goal of the ConEx protocol is that the important policy mechanisms can be implemented per logical link without per flow state (see Section 5.4). However, the price to pay can be flow state to audit ConEx signals (Section 5.5). This is justified in that i) auditing at the edges, with limited per flow state, enables policy elsewhere, including in the core, without any per flow state; ii) auditing can use soft flow state, which does not require route pinning.

There is a long standing argument over units of congestion: bytes vs packets (see [RFC7141] and its references). Section 4.6 explains why this problem must be addressed carefully. However, this document does not take a strong position on this issue. Nonetheless, it does require that the units of congestion must be an explicitly stated property of any proposed encoding, and the consequences of that design decision must be evaluated along with other aspects of the design.

To be successful the ConEx protocol needs to have the property that the relevant stakeholders each have the incentive to unilaterally start on each stage of partial deployment, which in turn creates incentives for further deployment. Furthermore, legacy systems that will never be upgraded do not become a barrier to deploying ConEx. Issues relating to partial deployment are described in Section 6.

Note that ConEx signals are not intended to be used for fine-grained congestion control. They are anticipated to be most useful at longer time scales and/or at coarser granularity than single microflows. For example the total congestion caused by a user might serve as an input to higher level policy or accountability functions, designed to create incentives for improving user behavior, such as choosing to send large quantities of data at off-peak times, at lower data rates or with less aggressive protocols such as LEDBAT [RFC6817] (see [RFC6789]).

Ultimately ConEx signals have the potential to provide a mechanism to regulate global Internet congestion. From the earliest days of congestion control research there has been a concern that there is no mechanism to prevent transport designers from incrementally making protocols more aggressive without bound and spiraling to a "tragedy of the commons" Internet congestion collapse. The "TCP friendly" paradigm was created in part to forestall this failure. However, it no longer commands any authority because it has little to say about the Internet of today, which has moved beyond the scaling range of standard TCP. As a consequence, many transports and applications are opening arbitrarily large numbers of connections or using arbitrary levels of aggressiveness. ConEx represents a recognition that the IETF cannot regulate this space directly because it concerns the behaviour of users and applications, not individual transport protocols. Instead the IETF can give network operators the protocol tools to arbitrate the space themselves, with better bulk traffic management. This in turn should create incentives for users, and designers of application and of transport protocols to be more mindful about contributing to congesting.

2.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

ConEx signals in IP packet headers from the sender to the network:

- Not-ConEx: The transport (or at least this packet) is not ConEx-capable.
- ConEx-Capable: The transport is ConEx-Capable. This is the opposite of Not-ConEx.
- ConEx Signal: A signal in a packet sent by a ConEx Capable transport. It carries at least one of the following signals:
  - Re-Echo-Loss: The transport has experienced a loss.
  - Re-Echo-ECN: The transport has detected an ECN congestion experienced (CE) mark.

Credit: The transport is building up credit to signal advance notice of the risk of packets contributing to congestion, in contrast to signalling only after inherently delayed feedback of actual congestion.

ConEx-Not-Marked: The transport is ConEx-capable but is signaling none of Re-Echo-Loss, Re-Echo-ECN or Credit.

ConEx-Marked: At least one of Re-Echo-Loss, Re-Echo-ECN or Credit.

ConEx-Re-Echo: At least one of Re-Echo-Loss or Re-Echo-ECN.

3. Requirements for the ConEx Abstract Mechanism

First time readers may wish to skim this section, since it is more understandable having read the entire document.

3.1. Requirements for ConEx Signals

Ideally, all the following requirements would be met by a Congestion Exposure Signal:

a. The ConEx Signal SHOULD be visible to internetwork layer devices along the entire path from the transport sender to the transport receiver. Equivalently, it SHOULD be present in the IPv4 or IPv6 header, and in the outermost IP header if using IP in IP tunneling. It MAY need to be visible if other encapsulating headers are used to interconnect networks. The ConEx Signal SHOULD be immutable once set by the transport sender. A corollary of these requirements is that the chosen ConEx encoding SHOULD pass silently without modification through pre-existing networking gear.

b. The ConEx Signal SHOULD be useful under only partial deployment. A minimal deployment SHOULD only require changes to transport senders. Furthermore, partial deployment SHOULD create incentives for additional deployment, both in terms of enabling ConEx on more devices and adding richer features to existing devices. Nonetheless, ConEx deployment need never be universal, and it is anticipated that some hosts and some transports may never support the ConEx Protocol and some networks may never use the ConEx Signals.

c. The ConEx signal SHOULD be timely. There will be a minimum delay of one RTT, and often longer if the transport protocol sends infrequent feedback (consider RTCP [RFC3550], [RFC6679] for example).

d. The ConEx signal SHOULD be accurate and auditable. The general approach for auditing is to observe the volume of congestion signals and ConEx signals on the forward data path and verify that the ConEx signals do not under-represent the congestion signals (see Section 5.5).

e. The ConEx signals for packet loss and ECN marking SHOULD have distinct encodings because they are likely to require different auditing techniques.
f. Additionally there SHOULD be an auditable ConEx Credit signal. A sender can use Credit to indicate potential future congestion, for example as often seen during startup. ConEx Credit is intended to overestimate congestion actually experienced across the network.

It is already known that implementing ConEx signals is likely to entail some compromises, and therefore all the requirements above are expressed with the keyword 'SHOULD' rather than 'MUST'. The only mandatory requirement is that a concrete protocol description MUST give sound reasoning if it chooses not to meet some requirement.

3.2. Constraints on the Audit Function

The role of the audit function and constraints on it are described in Section 5.5. There is no intention to standardise the audit function. However, it is necessary to lay down the following normative constraints on audit behaviour so that transport designers will know what to design against and implementers of audit devices will know what pitfalls to avoid:

Minimal False Hits: Audit SHOULD introduce minimal false hits for honest flows;

Minimal False Misses: Audit SHOULD quickly detect and sanction dishonest flows, ideally on the first dishonest packet;

Transport Oblivious: Audit SHOULD NOT be designed around one particular rate response, such as any particular TCP congestion control algorithm or one particular resource sharing regime such as TCP-friendliness [RFC5348]. An important goal is to give ingress networks the freedom to unilaterally allow different rate responses to congestion and different resource sharing regimes [Evol_cc], without having to coordinate with other networks over details of individual flow behaviour;

Sufficient Sanction: Audit SHOULD introduce sufficient sanction (e.g. loss in goodput) such that senders cannot gain from understating congestion;

Proportionate Sanction: To the extent that the audit might be subject to false hits, the sanction SHOULD be proportionate to the degree to which congestion is understated. If audit over-punishes, attackers will find ways to harness it into amplifying attacks on others. Ideally audit should, in the long-run, cause the user to get no better performance than they would get by being accurate.

Manage Memory Exhaustion: Audit SHOULD be able to counter state exhaustion attacks. For instance, if the audit function uses flow-state, it should not be possible for senders to exhaust its memory capacity by gratuitously sending numerous packets, each with a different flow ID.

Identifier Accountability: Audit SHOULD NOT be vulnerable to 'identity whitewashing', where a transport can label a flow with a new ID more cheaply than paying the cost of continuing to use its current ID [CheapPseud];

3.3. Requirements for non-abstract ConEx specifications

An experimental ConEx specification SHOULD describe the following protocol details:

Network Layer:

A. The specific ConEx signal encodings with packet formats, bit fields and/or code points;

B. An inventory of invalid combinations of flags or invalid codepoints in the encoding. Whether security gateways should normalise, discard or ignore such invalid encodings, and what values they should be considered equivalent to by ConEx-aware elements;

C. An inventory of any conflated signals or any other effects that are known to compromise signal integrity;

D. Whether the source is responsible for allowing for the round trip delay in ConEx signals (e.g. using a Credit marking), and if so whether Credit is maintained for the duration of a flow or degrades over time, and what defines the end of the duration of a flow;

E. A specification for signal units (bytes vs packets, etc), any approximations allowed and algorithms to do any implied conversions or accounting;

F. If the units are bytes a definition of which headers are included in the size of the packet;

G. How tunnels should propagate the ConEx encoding;

H. Whether the encoding fields are mutable or not, to ensure that header authentication, checksum calculation, etc. process them correctly. A ConEx encoding field SHOULD be immutable end-to-end, then end points can detect if it has been tampered with in transit;

I. If a specific encoding allows mutability (e.g. at proxies), an inventory of invalid transitions between codepoints. In all encodings, transitions from any ConEx marking to Not-ConEx MUST be invalid;

J. A statement that the ConEx encoding is only applicable to unicast and anycast, and that forwarding elements should silently ignore any ConEx signalling on multicast packets (they should be forwarded unchanged)

K. Definition of any extensibility;
L. Backward and forward compatibility and potential migration strategies. In all cases, a ConEx encoding MUST be arranged so that legacy transport senders implicitly send Not-ConEx;
M. Any (optional) modification to data-plane forwarding dependent on the encoding (e.g. preferential discard, interaction with Diffserv, ECN etc.);
N. Any warning or error messages relevant to the encoding.

Note regarding item J on multicast: A multicast tree may involve different levels of congestion on each leg. Any traffic management can only monitor or control multicast congestion at or near each receiver. It would make no sense for the sender to try to expose "whole path congestion" in sent packets, because it cannot hope to describe all the differing congestion levels on every leg of the tree.

Transport Layer:

A. A specification of any required changes to congestion feedback in particular transport protocols.
B. A specification (or minimally a recommendation) for how a transport should estimate credits at the beginning of a connection and while it is in progress.
C. A specification of whether any other protocol options should (or must) be enabled along with an implementation of ConEx (e.g. at least attempting to negotiate ECN and SACK capability);
D. A specification of any configuration that a ConEx stack may require (or preferably confirmation that it requires no configuration);
E. A specification of the statistics that a protocol stack should log for each type of marking on a per-flow or aggregate basis.

Security:

A. An example of a strong audit algorithm suitable for detecting if a single flow is misstating congestion. This algorithm should present minimal false results, but need not have optimal scaling properties (e.g. may need per flow state).
B. An example of an audit algorithm suitable for detecting misstated congestion in a large aggregate (e.g. no per-flow state).

The possibility exists that these specifications over constrain the ConEx design, and can not be fully satisfied. An important part of the evaluation of any particular design will be a thorough inventory of all ways in which it might fail to satisfy these specifications.

4. Encoding Congestion Exposure

Most protocol specifications start with a description of packet formats and codepoints with their associated meanings. This document does not: It is already known that choosing the encoding for ConEx is likely to entail some engineering compromises that have the potential to reduce the protocol's usefulness in some settings. For instance the experimental ConEx encoding chosen for IPv6 [I-D.ietf-conex-destopt] had to make compromises on tunnelling. Rather than making these engineering choices prematurely, this document sidesteps the encoding problem by making it abstract. It describes several different representations of ConEx Signals, none of which are specified to the level of specific bits or code points.

The goal of this approach is to be as complete as possible for discovering the potential usage and capabilities of the ConEx protocol, so we have some hope of making optimal design decisions when choosing the encoding. Even if experiments reveal particular problems due to the encoding, then this document will still serve as a reference model.

4.1. Naive Encoding

For tutorial purposes, it is helpful to describe a naive encoding of the ConEx protocol for TCP and similar protocols: set a bit (not specified here) in the IP header on each retransmission and on each ECN signaled window reduction. Network devices along the forward path can see this bit and act on it. For example any device along the path might limit the rate of all traffic if the rate of marked (congested) packets exceeds a threshold.

This simple encoding is sufficient to illustrate many of the benefits envisioned for ConEx. At first glance it looks like it might motivate people to deploy and use it. It is a one line code change that a small number of OS developers and content providers could unilaterally deploy across a significant fraction of all Internet traffic. However, this encoding does not support auditing so it would also motivate users and/or applications to misrepresent the congestion that they are causing [RFC3514]. As a consequence the naive encoding is not likely to be trusted and thus creates its own disincentives for deployment.

Nonetheless, this Naive encoding does present a clear mental model of how the ConEx protocol might function under various uses. It is useful for thought experiments where it can be stipulated that all participants are honest and it does illustrate some of the incentives that might be introduced by ConEx.

4.2. Null Encoding

In limited contexts it is possible to implement ConEx-like functions without any signals at all by measuring rest-of-path congestion directly from TCP headers. The algorithm is to keep at least one RTT of past TCP headers and matching each new header against the history to count duplicate data.

This could implement many ConEx policies, without any explicit protocol. It is fairly easy to implement, at least at low rate (e.g. in a software based edge router). However, it would only be useful in cases where the network operator can see the TCP headers. This is currently (2014) the majority of traffic because UDP, IPSec and VPN tunnels are used far less than SSL or TLS over TCP/IP, which do not hide TCP sequence numbers from network devices. However, anyone specifically intending to avoid the attention of a congestion policy device would only have to hide their TCP headers from the network operator (e.g. by using a VPN tunnel).

4.3. ECN Based Encoding

The re-ECN specification [I-D.briscoe-conex-re-ecn-tcp] presents an encoding of ConEx in IPv4 and IPv6 that was tightly integrated with ECN encoding in order to fit into the IPv4 header. Any individual packet may need to represent any ECN codepoint and any ConEx signal value independently. So, ideally their encoding should be entirely independent. However, given the limited number of header bits and/or code points, re-ECN chooses to partially share code points and to re-echo both losses and ECN with just one codepoint.

The central theme of the re-ECN work is an audit mechanism that provides sufficient disincentives against misrepresenting congestion [I-D.briscoe-conex-re-ecn-motiv]. It is analyzed extensively in Briscoe's PhD dissertation [Refb-dis]. For a tutorial background on re-ECN motivation and techniques, see [Re-fb, FairerFaster].

Re-ECN is an example of one chosen set of compromises attempting to meet the requirements of Section 3. The present document takes a step back, aiming to state the ideal requirements in order to allow the Internet community to assess whether different compromises might be better.

The problem with Re-ECN is that it requires that receivers be ECN enabled in addition to sender changes. Newer encodings [I-D.ietf-conex-destopt] overcome this problem by being able to represent loss and ECN based congestion separately.

4.4. Independent Bits

This encoding involves flag bits, each of which the sender can set independently to indicate to the network one of the following four signals:

ConEx (Not-ConEx) The transport is (or is not) using ConEx with this packet (network layer encoding requirement L in Section 3.3) says the protocol must be arranged so that legacy transport senders implicitly send Not-ConEx;

Re-Echo-Loss (Not-Re-Echo-Loss) The transport has (or has not) experienced a loss

Re-Echo-ECN (Not-Re-Echo-ECN) The transport has (or has not) experienced ECN-signaled congestion

Credit (Not-Credit) The transport is (or is not) building up congestion credit (see Section 5.5 on the audit function)

A packet with ConEx set combined with all the three other flags cleared implies ConEx-Not-Marked

This encoding does not imply any exclusion property among the signals. Multiple types of congestion (ECN, loss) can be signalled on the same ACK. So, ideally, a ConEx sender would be able to reflect these in the next packet. However, there will be many invalid combinations of flags (e.g. Not-ConEx combined with any of the ConEx-marked flags), which a malicious sender could use to advantage against naive policy devices that only check each flag separately.

As long as the packets in a flow have uniform sizes, it does not matter whether the units of congestion are packets or bytes. However, if an application sends very irregular packet sizes, it may be necessary for the sender to mark multiple packets to avoid being in technical violation of an audit function measuring in bytes (see Section 4.6).

4.5. Codepoint Encoding

This encoding involves signaling one of the following five codepoints:

ENUM {Not-ConEx, ConEx-Not-Marked, Re-Echo-Loss, Re-Echo-ECN, Credit}

Each named codepoint has the same meaning as in the encoding using independent bits in the previous section. The use of any one codepoint implies the negative of all the others.

Inherently, the semantics of most of the enumerated codepoints are mutually exclusive. 'Credit' is the only one that might need to be used in combination with either Re-Echo-Loss or Re-Echo-ECN, but even that requirement is questionable. It must not be forgotten that the enumerated encoding loses the flexibility to signal these two combinations, whereas the encoding with four independent bits is not so limited. Alternatively two extra codepoints could be assigned to these two combinations of semantics. The comment in the previous section about units also applies.

4.6. Units Implied by an Encoding

The following comments apply generally to all the other encodings.

Congestion can be due to exhaustion of bit-carrying capacity, or exhaustion of packet processing power. When a packet is discarded or marked to indicate congestion, there is no easy way to know whether the lost or marked packet signifies bit-congestion or packet-congestion. The above ConEx encodings that rely on marking packets suffer from the same ambiguity.

This problem is most acute when audit needs to check that one count of markings matches another. For example if there are ConEx markings on three large (1500B) packets, is that sufficient to match the loss of 5 small (60B) packets? If a packet-marking is defined to mean all the bytes in the packet are marked, then we have 4500B of Conex marked data against 300B of lost data, which is easily sufficient. If instead we are counting packets, then we have 3 ConEx packets against 5 lost packets, which is not sufficient. This problem will not arise when all the packets in a flow are the same size, but a choice needs to be made for flows in which packet sizes vary, such as BGP, SPDY and some variable rate video encoding schemes.

Whether to use bytes or packets is not obvious. For instance, the most expensive links in the Internet, in terms of cost per bit, are all at lower data rates, where transmission times are large and packet sizes are important. In order for a policy to consider wire time, it needs to know the number of congested bytes. However, high speed networking equipment and the transport protocols themselves sometimes gauge resource consumption and congestion in terms of packets.

[RFC7141] advises that congestion indications should be interpreted in units of bytes when responding to congestion, at least on today’s Internet. [RFC6789] takes the same view in its definition of congestion-volume, again for today’s Internet.

In any TCP implementation this is simple to achieve for varying size packets, given TCP SACK tracks losses in bytes. If an encoding is specified in units of bytes, the encoding should also specify which headers to include in the size of a packet (see network layer requirement F in Section 3.3).

RFC 7141 constructs an argument for why equipment tends to be built so that the bottleneck will be the bit-carrying capacity of its interfaces not its packet processing capacity. However, RFC 7141 acknowledges that the position may change in future, and notes that new techniques will need to be developed to distinguish packet- and bit-congestion.

Given this document describes an abstract ConEx mechanism, it is intended to be timeless. Therefore it does not take a strong position on this issue. However, a ConEx encoding will need to explicitly specify whether it assumes units of bytes or packets consistently for both congestion indications and ConEx markings (see network layer requirement E in Section 3.3). It may help to refer to the guidance in [RFC7141].

5. Congestion Exposure Components

The components shown in Figure 1 as well as policy and audit are described in more detail.

5.1. Network Devices (Not modified)

Congestion signals originate from network devices as they do today. A congested router, switch or other network device can discard or ECN mark packets when it is congested.

5.2. Modified Senders

The sending transport needs to be modified to send Congestion Exposure signals in response to congestion feedback signals (e.g. for the case of a TCP transport see [I-D.ietf-tcp-modifications]). We want to permit ConEx without ECN (e.g. if the receiver does not support ECN). However, we want to encourage a ConEx sender to at least attempt to negotiate ECN (a ConEx transport protocol spec may require this), because it is believed that ConEx without ECN is harder to audit, and thus potentially exposed to cheating. Since honest users have the potential to benefit from stronger mechanisms to manage traffic they have an incentive to deploy ConEx and ECN together. This incentive is not sufficient to prevent a dishonest user from constructing (or configuring) a sender that enables ConEx after choosing not to negotiate ECN, but it should be sufficient to prevent this from being the sustained default case for any significant pool of users.

Permitting ConEx without ECN is necessary to facilitate bootstrapping other parts of ConEx deployment.

5.3. Receivers (Optionally Modified)

Any receiving transport may already feedback sufficiently useful signals to the sender so that it does not need to be altered.

The native loss or ECN signaling mechanism required for compliance with existing congestion control standards (e.g. RTCP, SCTP) will typically be sufficient for the Sender to generate ConEx signals.

TCP’s loss feedback is sufficient for ConEx if SACK is used [RFC2018]. However, the original specification for ECN in TCP [RFC3168] signals congestion no more than once per round trip. The sender may require more precise feedback from the receiver otherwise it is at risk of appearing to be understating its ConEx Signals.

Ideally, ConEx should be added to a transport like TCP without mandatory modifications to the receiver. But in the TCP-ECN case an optional modification to the receiver could be recommended for precision (see [I-D.ietf-tcpm-accecn-reqs], which is based on the approach originally taken when adding re-ECN to TCP [I-D.briscoe-conex-re-ecn-tcp]).

5.4. Policy Devices

Policy devices are characterised by a need to be configured with a policy related to the users or neighboring networks being served. In contrast, auditing devices solely enforce compliance with the ConEx protocol and do not need to be configured with any client-specific policy.

One of the design goals of the ConEx protocol is that none of the important policy mechanisms requires per flow state, and that policy mechanisms can even be implemented for heavily aggregated traffic in the core of the Internet with complexity akin to accumulating marking volumes per logical link. Of course, policy mechanisms may sometimes choose to focus down on individual flows, but ConEx aims to make aggregate policy devices feasible.

5.4.1. Congestion Monitoring Devices

Policy devices can typically be decomposed into two functions i) monitoring the ConEx signal to compare it with a policy then ii) acting in some way on the result. Various actions might be invoked against ’out of contract’ traffic, such as policing (see Section 5.4.3), re-routing, or downgrading the class of service.

Alternatively a policy device might not act directly on the traffic, but instead report to management systems that are designed to control congestion indirectly. For instance the reports might trigger capacity upgrades, penalty clauses in contracts, levy charges based on congestion, or merely send warnings to clients who are causing excessive congestion.

Nonetheless, whatever action is invoked, the congestion monitoring function will always be a necessary part of any policy device.

5.4.2. Rest-of-Path Congestion Monitoring

ConEx signals indicate the level of congestion along a whole path from source to destination. In contrast, ECN signals monitored in the middle of a network indicate the level of congestion experienced so far on the path (of course, only in ECN-capable traffic).

If a monitor in the middle of a network (e.g. at a network border) measures both of these signals, it can subtract the level of ECN (path so far) from the level of ConEx (whole path) to derive a measure of the congestion that packets are likely to experience between the monitoring point and their destination (rest-of-path congestion).

It will often be preferable for policy devices to monitor rest-of-path congestion if they can, because it is a measure of the downstream congestion that the policy device can directly influence by controlling the traffic passing through it.

5.4.3. Congestion Policers

A congestion policer can be implemented in a very similar way to a bit-rate policer, but its effect can be focused solely on traffic of users causing congestion downstream, which ConEx signals make visible. Without ConEx signals, the only way to mitigate congestion is to blindly limit traffic bit-rate, on the assumption that high bit-rate is more likely to cause congestion.

A congestion policer monitors all ConEx traffic entering a network, or some identifiable subset. Using ConEx signals and/or Credit signals (and preferably subtracting ECN signals to yield rest-of-path congestion), it measures the amount of congestion that this traffic is contributing somewhere downstream. If this persistently exceeds a policy-configured 'congestion-bit-rate' the congestion policer can limit all the monitored ConEx traffic.

A congestion policer can be implemented by a simple token bucket applied to an aggregate. But unlike a bit-rate policer, it removes tokens only when it forwards packets that are ConEx-Marked and/or Credit-Marked, effectively treating Not-ConEx-Marked packets as invisible. Consequently, because tokens give the right to send congested bits, the fill-rate of the token bucket will represent the allowed congestion-bit-rate. This should provide sufficient traffic management without having to additionally constrain the straight bit-rate at all. See [I-D.briscoe-conex-policing] for details.

Note that the policing action could be to introduce a throttle (discard some traffic) immediately upstream of the congestion monitor. Alternatively, this throttle could introduce delay using a queue with its own AQM, which potentially increases the whole path congestion. In effect the congestion policer has moved the congestion earlier in the path, and focused it on one user to protect downstream resources by reducing the congestion in the rest of the path.

5.5. Audit

The most critical aspect of ConEx is the capability to support robust auditing. It can be assumed that sanctions based on ConEx signals will create an intrinsic motivation for users to understate the congestion that they are causing. So, without strong audit functions, the ConEx signal would become understated to the point of being useless. Therefore the most important feature of an encoding design is likely to be the robustness of the auditing it supports.

The general goal of an auditor is to make sure that any ConEx-enabled traffic is sent with sufficient ConEx-Re-Echo and ConEx-Credit signals. A concrete definition of the ConEx protocol MUST define what sufficient means.

If a ConEx-enabled transport does not carry sufficient ConEx signals, then an auditor is likely to apply some sanction to that traffic. Although sanctions are beyond the scope of this document, an example sanction might be to throttle the traffic immediately upstream of the auditor to prevent the user from getting any advantage by understating congestion. Such a throttle would likely include some combination of delaying or dropping traffic.

A ConEx auditor might use one of the following techniques:

Generic loss auditing: For congestion signaled by loss, totally accurate auditing is not believed to be possible in the general case, because it involves a network node detecting the absence of some packets, when it cannot always necessarily identify retransmissions or missing packets. The missing packet might simply be taking a different route, or the IP payload may be encrypted.

It is for this reason that it is desirable to motivate the deploying of ECN, even though ECN is not strictly required for ConEx.

ECN auditing: Directly observe and compare the volume of ECN and ConEx marks. Since the volume of ECN marks rises monotonically along a path, ECN auditing is most accurate when located near the transport receiver. For this reason ECN should be monitored downstream of the predominant bottleneck.

TCP-specific loss auditing: For non-encrypted standard TCP traffic on a single path, a tactical audit approach could be to measure losses by detecting retransmissions, which appear as duplicate sequence numbers upstream of the loss and out of order data downstream of the loss. Since some reordering is present in the Internet, such a loss estimator would be most accurate near the sender. Such an audit device should treat non-ECN-capable packets with encrypted IP payload as Not-ConEx, even if they claim to be ConEx-capable, unless the operator is also using one of the other two techniques below that can audit such packets against losses.

Predominant bottleneck loss auditing: For networks designed so that losses predominantly occur under the control of one IP-aware bottleneck node on the path, the auditor could be located at this bottleneck. It could simply compare ConEx Signals with actual local packet discards (and ECN marks). This is a good model for most consumer access networks where audit accuracy could well be sufficient even if losses occasionally occur elsewhere in the network.

Although the auditor at the predominant bottleneck would not be able to count losses at other nodes, transports would not know where losses were occurring either. Therefore a transport would not know which losses it could cheat and which ones it couldn't without getting caught.

ECN tunnel loss auditing: A network operator can arrange IP-in-IP tunnels (or IP-in-MPLS etc.) so that any losses within the tunnels are deferred until the tunnel egress. Then the audit function can be deployed at the egress and be aware of all losses. This is possible by enabling ECN marking on switches and routers within a tunnel, irrespective of whether end-systems support ECN, by exploiting a side-effect of the way tunnels handle the ECN field. After encapsulation at the tunnel ingress, the network should arrange for any non-ECN packets (with '00' in ECN field of the outer) to be set to the ECN-capable transport (ECT(0)) codepoint.

Then, if they experience congestion at one of the ECN-capable switches or routers within the tunnel, some will be ECN-marked rather than immediately dropped. However, when the tunnel decapsulator strips the outer from such an ECN-marked packet, if it finds the inner header has '00' in the ECN field (meaning that the endpoints do not support ECN) it will automatically drop the packet, assuming it complies with [RFC6040]. Thus, an audit function at the decapsulator can know which packets would have been dropped within the tunnel (and even which are genuinely ECN-marked for the end-to-end protocol). Non-ECN end-systems outside the tunnel see no sign of the use of ECN internally.

In addition, other audit techniques may be identified in the future.

[Refb-dis] gives a comprehensive inventory of attacks against audit proposed by various people. It includes pseudocode for both deterministic and statistical audit functions designed to thwart these attacks and analyses the effectiveness of an implementation. Although this work is specific to the re-ECN protocol, most of the material is useful for designing and assessing audit of other specific ConEx encodings, against both ECN and loss.

The auditing function should be able to trigger sufficient sanction to discourage understating congestion [Salvatori05]. This seems to require designing the sanction in concert with the policy functions, even though they might be implemented in different parts of the network. However, [Refb-dis] proves audit and policy functions can be independent as long as audit drops sufficient traffic to 'normalise' actual congestion signals to be no greater than ConEx signals.

Similarly, the job of incentivising the sending of ConEx-enabled packets is proper solely to policy devices, independent of the audit function. The audit function's job is policy-neutral, so it should be solely confined to checking for correctness within those packets that have been marked as ConEx-capable. Even if there are Not-ConEx packets mixed with ConEx packets within a flow, audit will not need to monitor any Not-ConEx packets.

Note that in the future it might prove to be desirable to provide advice on uniformly implementing sanctions, because otherwise insufficient sanctions could impair the ability to implement policy elsewhere in the network.

Some of the audit algorithms require per flow state. This cost is expected to be tolerable, because these techniques are most apropos near the edges of the network, where traffic is generally much less aggregated, so the state need not overwhelm any one device. The flow-state required for audit creates itself as it detects new flows. Therefore a flow will not fail if it is re-routed away from the audit box currently holding its flow-state, so auditing does not require route pinning and works fine with multipath flows.

Holding flow-state seems to create a vulnerability to attacks that exhaust the auditor's memory by opening numerous new short flows. The audit function can protect itself from this attack by not allocating new flow-state unless a ConEx-marked packet arrives (e.g. credit at the start of a flow). Because policy devices rate limit ConEx-marked packets, this sets a natural limit to the rate at which a source can create flow-state in audit devices. The auditor would treat all the remaining flows without any ConEx-marked packets as a single misbehaving aggregate.

Auditing can be distributed and redundant. One flow may be audited in multiple places, using multiple techniques. Some audit techniques do not require any per flow state and can be applied to aggregate traffic. These might be able to detect the presence of understated congestion at large scale and support recursively hunting for individual flows that are understating their congestion. Even at large scales, flows can be randomly selected for individual auditing.

Sampling techniques can also be used to bound the total auditing memory footprint, although the implementer needs to counter the tactic where a source cheats until caught by sampling, then simply discards that flow ID and starts cheating with a new one (termed 'identifier white-washing when caught').

For the the concrete ConEx protocol encoding defined in [I-D.ietf-conex-destopt], ConEx Credit and ConEx-Re-Echo signals are intended to be audited separately. The Credit signal can be audited directly against actual congestion (loss and ECN). However, there will be an inherent delay of at least one round trip between a congestion signal and the subsequent ConEx-Re-Echo signal it triggers, as shown in Figure 1. Therefore ConEx-Re-Echo signals will need to be audited with some allowance for this delay. Further discussion of design and implementation choices for functions intended to audit this concrete ConEx encoding can be found in [I-D.wagner-conex-audit].

6. Support for Incremental Deployment

The ConEx abstract protocol described so far is intended to support incremental deployment in every possible respect. For convenience, the following list collects together all the features that support incremental deployment in the concrete ConEx specifications, and points to further information on each:

Packets: The wire protocol encoding allows each packet to indicate whether it is using ConEx or not (see Section 4 on Encoding Congestion Exposure).

Senders: ConEx requires a modification to the source in order to send ConEx packet markings (see Section 5.2). Although ConEx support can be indicated on a packet-by-packet basis, it is likely that all the packets in a flow will either consistently support ConEx or consistently not. It is also likely that, if the implementation of a transport protocol supports ConEx, all the packets sent from that host using that protocol will be ConEx marked.

The implementations of some of the transport protocols on a host might not support ConEx (e.g. the implementation of DNS over UDP might not support ConEx, while perhaps RTP over UDP and TCP will). Any non-upgraded transports and non-upgraded hosts will simply continue to send regular Not-ConEx packets as always.

A network operator can create incentives for senders to voluntarily reveal ConEx information (see the item on incremental deployment by 'Networks' below).

Receivers: A ConEx source should be able to work with the regular receiver for the transport in question, without requiring any ConEx-specific modifications. This is true for modern transport protocols (RTCP, SCTP etc) and it is even true for TCP, as long as the receiver supports SACK, which is widely deployed anyway. However, it is not true for ECN feedback in TCP. The need for more precise ECN feedback in TCP is not exclusive to ConEx, for instance Data Centre TCP (DCTCP [DCTCP]) uses precise feedback to good effect. Therefore, if a receiver offers precise feedback, [I-D.ietf-tcpm-accecn-reqs] it will be best if ConEx uses it (see Section 5.3). Alternatively, without sufficiently precise congestion feedback from the receiver, the source may have to conservatively send extra ConEx markings in order to avoid understating congestion.

Proxies: Although it was stated above that ConEx requires a modification to the source, ConEx signals could theoretically be introduced by a proxy for the source, as long as it can intercept feedback from the receiver. Similarly, more precise feedback could thoretically be provided by a proxy for the receiver rather than modifying the receiver itself.

Forwarding: No modification to forwarding or queuing is needed for ConEx.

However, once some ConEx is deployed, it is possible that a queue implementation could optionally take advantage of the ConEx information in packets. For instance, it has been suggested [I-D.ietf-conex-destopt] that a queue would be more robust against flooding if it preferentially discarded Not-ConEx packets then Not-Marked ConEx packets.

A ConEx sender re-echoes congestion whether the queues signaling congestion are ECN-enabled or not. Nonetheless, an operator relying on ConEx signals is recommended to enable ECN in queues wherever possible. This is because auditing works best if most congestion is indicated by ECN rather than loss (see Section 3). Also, monitoring rest-of-path congestion is not accurate if there are congested non-ECN queues upstream of the monitoring point (Section 5.4.2).

Networks: If a subset of traffic sources (or proxies) use ConEx signals to reveal congestion in the internetwork layer, a network operator can choose (or not) to use this information for traffic management. As long as the end-to-end ConEx signals are present, each network can unilaterally choose to use them--independently of whether other networks do.

ConEx marked packets may safely traverse a network that ignores them. ConEx signals are defined to remain unchanged once set by the sender, but some encodings may allow changes in transit (e.g. by proxies). In no circumstances will a network node change ConEx marked packets to Not-ConEx (network layer encoding requirement I in Section 3.3). If necessary, endpoints should be able to detect if a network is removing ConEx signals (network layer encoding requirement H in Section 3.3).

An operator can deploy policy devices (Section 5.4) wherever traffic enters its network, in order to monitor the downstream congestion that incoming traffic contributes to, and control it if necessary. A network operator can create incentives for the developers of sending applications and transports to voluntarily reveal ConEx information. Without ConEx information, a network operator tends to have to limit the bit-rate or volume from a site more than is necessary, just in case it might congest others. With ConEx information, the operator can solely limit congestion-causing traffic, and otherwise allow complete freedom. This greater freedom acts as an inducement for the source to volunteer ConEx information. An operator may also monitor whether a source transport has sent ConEx packets, and treat the same transport with greater suspicion (e.g. a more stringent rate-limit) whenever it selectively sends packets without ConEx support. See [RFC6789] for further discussion of deployment incentives for networks and references to scenarios where some networks use ConEx-based policy devices and others don't.

An operator can deploy audit devices (Section 5.5) unilaterally within its own network to verify that traffic sources are not understating ConEx information. From the viewpoint of one network operator (say N_a), it only cares that the level of ConEx signaling is sufficient to cover congestion in its own network. If traffic continues into a congested downstream network (say N_b), it is of no concern to the first network (N_a) if the end-to-end ConEx signaling is insufficient to cover the congestion in N_b as well. This is N_b's concern, and N_b can both detect such anomalous traffic and deal with it using ConEx-based audit devices itself.

7. IANA Considerations

This memo includes no request to IANA.

Note to RFC Editor: this section may be removed on publication as an RFC.

8. Security Considerations

The only known risk associated with ConEx is that users and applications are very likely to be motivated to under-represent the congestion that they are causing. Significant portions of this document are about mechanisms to audit the ConEx signals and create sufficient sanction to inhibit such under-representation. In particular see Section 5.5.

Security attacks and their defences are best discussed against a concrete protocol specification, not the abstract mechanism of this document. A concrete ConEx protocol will need to be accompanied by a document describing how the protocol and its audit mechanisms defend against likely attacks. [Refb-dis] will be a useful source for such a document. It gives a comprehensive inventory of attacks against audit that have been proposed by various parties. It includes pseudocode for both deterministic and statistical audit functions designed to thwart these attacks and analyses the effectiveness of an implementation.

However, [Refb-dis] is specific to the re-ECN protocol, which signalled ECN & loss together, whereas the concrete ConEx protocol defined in [I-D.ietf-conex-destopt] signals them separately.

Therefore, although likely attacks will be similar, there will be more combinations of attacks to worry about, and defences and their analysis are likely to be a little different for ConEx.

The main known attacks that a security document for a concrete ConEx protocol will need to address are listed below, and [Refb-dis] should be referred to for how re-ECN was designed to defend against similar attacks:

- Attacks on the audit function (see Section 7.5 of [Refb-dis]):
  - Flow ID Whitewashing: Designing the audit function so that a source cannot gain from starting a new flow once audit has detected cheating in a previous flow.
  - Dragging Down an Aggregate: Avoiding audit discarding packets from all flows within an aggregate, which would allow one flow to pull down the average so that the audit function would discard packets from all flows, not just the offending flow.
  - Dragging Down a Spoofed Flow ID: An attacker understates ConEx markings in packets that spoof another flow, which fools the audit function into dropping the genuine user's packets.
- Attacks by networks on other networks (see Section 8.2 of [Refb-dis]):
  - Dummy Traffic: Sending dummy traffic across a border with understated ConEx markings to bring down the average ConEx markings in the aggregate of border traffic. This attack can be combined with a TTL that expires before the packets reach an audit function.
  - Signal Poisoning with 'Cancelled' Marking: Sending high volumes of valid packets that are both ConEx-Marked and ECN-Marked, which seems to represent congestion upstream, but it makes these packets immune to being further ECN-Marked downstream.

It is planned to document all known attacks and their defences (including all the above) in the RFC series against a concrete ConEx protocol specification. In the interim [Refb-dis] and its references should be referred to for details and ways to address these attacks in the case of re-ECN.

9. Acknowledgements

This document was improved by review comments from Toby Moncaster, Nandita Dukkipati, Mirja Kuehlewind, Caitlin Bestler, Marcelo Bagnulo Braun, John Leslie, Ingemar Johansson and David Wagner.

Bob Briscoe's work on this specification received part-funding from the European Union's Seventh Framework Programme FP7/2007-2013 under Trilogy 2 project, grant agreement no. 317756. The views expressed here are solely those of the author.

10. Comments Solicited

Comments and questions are encouraged and very welcome. They can be addressed to the IETF Congestion Exposure (ConEx) working group mailing list <conex@ietf.org>, and/or to the authors.

11. References

11.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

11.2. Informative References

[CheapPseud] Friedman, E. and P. Resnick, "The Social Cost of Cheap Pseudonyms", Journal of Economics and Management Strategy 10(2)173--199, 1998.

[DCTCP] Alizadeh, M., Greenberg, A., Maltz, D., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., and M. Sridharan, "Data Center TCP (DCTCP)", ACM SIGCOMM CCR 40(4)63--74, October 2010, <http://portal.acm.org/citation.cfm?id=1851192>.

[Evol_cc] Gibbens, R. and F. Kelly, "Resource pricing and the evolution of congestion control", Automatica 35(12)1969--1985, December 1999, <http://www.sciencedirect.com/science/article/pii/S0005109899001351>.

[FairerFaster] Briscoe, B., "A Fairer, Faster Internet Protocol", IEEE Spectrum Dec 2008:38--43, December 2008, <http://bobbriscoe.net/projects/refb/#fairfastip>.

[I-D.briscoe-conex-policing] Briscoe, B., "Network Performance Isolation using Congestion

Policing", draft-briscoe-conex-policing-00 (work in progress), February 2013.

[I-D.briscoe-conex-re-ecn-motiv] Briscoe, B., Jacquet, A., Moncaster, T., and A. Smith, "Re-ECN: A Framework for adding Congestion Accountability to TCP/IP", draft-briscoe-conex-re-ecn-motiv-02 (work in progress), July 2013.

[I-D.briscoe-conex-re-ecn-tcp] Briscoe, B., Jacquet, A., Moncaster, T., and A. Smith, "Re-ECN: Adding Accountability for Causing Congestion to TCP/IP", draft-briscoe-conex-re-ecn-tcp-02 (work in progress), July 2013.

[I-D.ietf-conex-destopt] Krishnan, S., Kuehlewind, M., and C. Ucendo, "IPv6 Destination Option for ConEx", draft-ietf-conex-destopt-05 (work in progress), October 2013.

[I-D.ietf-tcp-modifications] Kuehlewind, M. and R. Scheffenegger, "TCP modifications for Congestion Exposure", draft-ietf-conex-tcp-modifications-04 (work in progress), July 2013.

[I-D.ietf-tcpm-accecn-reqs] Kuehlewind, M. and R. Scheffenegger, "Problem Statement and Requirements for a More Accurate ECN Feedback", draft-ietf-tcpm-accecn-reqs-04 (work in progress), October 2013.

[I-D.wagner-conex-audit] Wagner, D. and M. Kuehlewind, "Auditing of Congestion Exposure (ConEx) signals", draft-wagner-conex-audit-01 (work in progress), February 2014.

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and A. Romanow, "TCP Selective Acknowledgment Options", RFC 2018, October 1996.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC3514] Bellovin, S., "The Security Flag in the IPv4 Header", RFC 3514, April 1 2003.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC5348] Floyd, S., Handley, M., Padhye, J., and J. Widmer, "TCP Friendly Rate Control (TFRC): Protocol Specification", RFC 5348, September 2008.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.

[RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, November 2010.

[RFC6679] Westerlund, M., Johansson, I., Perkins, C., O'Hanlon, P., and K. Carlberg, "Explicit Congestion Notification (ECN) for RTP over UDP", RFC 6679, August 2012.

[RFC6789] Briscoe, B., Woundy, R., and A. Cooper, "Congestion Exposure (ConEx) Concepts and Use Cases", RFC 6789, December 2012.

[RFC6817] Shalunov, S., Hazel, G., Iyengar, J., and M. Kuehlewind, "Low Extra Delay Background Transport (LEDBAT)", RFC 6817, December 2012.

[RFC7141] Briscoe, B. and J. Manner, "Byte and Packet Congestion Notification", BCP 41, RFC 7141, February 2014.

[Re-fb] Briscoe, B., Jacquet, A., Di Cairano-Gilfedder, C., Salvatori, A., Soppera, A., and M. Koyabe, "Policing Congestion Response in an Internetwork Using Re-Feedback", ACM SIGCOMM CCR 35(4)277--288, August 2005, <http://portal.acm.org/citation.cfm?id=1080091.1080124>.

[Refb-dis] Briscoe, B., "Re-feedback: Freedom with Accountability for Causing Congestion in a Connectionless Internetwork", UCL PhD Dissertation , 2009, <http://discovery.ucl.ac.uk/16274/>.

[Salvatori05] Salvatori, A., "Closed Loop Traffic Policing", Politecnico Torino and Institut Eurecom Masters Thesis , September 2005.

Authors' Addresses

Matt Mathis
Google, Inc
1600 Amphitheater Parkway
Mountain View, California 93117
USA

EMail: mattmathis at google.com

Bob Briscoe
BT
B54/77, Adastral Park
Martlesham Heath
Ipswich IP5 3RE
UK

Phone: +44 1473 645196
EMail: bob.briscoe@bt.com
URI: http://bobbriscoe.net/

ConEx B. Briscoe, Ed.
Internet-Draft BT
Intended status: Informational R. Woundy, Ed.
Expires: January 18, 2013 Comcast
A. Cooper, Ed.
CDT
July 17, 2012

ConEx Concepts and Use Cases
draft-ietf-conex-concepts-uses-05

Abstract

This document provides the entry point to the set of documentation about the Congestion Exposure (ConEx) protocol. It explains the motivation for including a ConEx marking at the IP layer: to expose information about congestion to network nodes. Although such information may have a number of uses, this document focuses on how the information communicated by the ConEx marking can serve as the basis for significantly more efficient and effective traffic management than what exists on the Internet today.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Concepts 5
  2.1. Congestion 5
  2.2. Congestion-Volume 5
  2.3. Rest-of-Path Congestion 6
  2.4. Definitions 6
3. Core Use Case: Informing Traffic Management 7
  3.1. Use Case Description 8
  3.2. Additional Benefits 9
  3.3. Comparison with Existing Approaches 9
4. Other Use Cases 11
5. Deployment Arrangements 12
6. Experimental Considerations 13
7. Security Considerations 14
8. IANA Considerations 14
9. Acknowledgments 14
  9.1. Contributors 15
10. Informative References 15

1. Introduction

The power of Internet technology comes from multiplexing shared capacity with packets rather than circuits. Network operators aim to provide sufficient shared capacity, but when too much packet load meets too little shared capacity, congestion results. Congestion appears as either increased delay, dropped packets or packets explicitly marked with Explicit Congestion Notification (ECN) markings [RFC3168]. As described in Figure 1, congestion control currently relies on the transport receiver detecting these 'Congestion Signals' and informing the transport sender in 'Congestion Feedback Signals.' The sender is then expected to reduce its rate in response.

This document provides the entry point to the set of documentation about the Congestion Exposure (ConEx) protocol. It focuses on the motivation for including a ConEx marking at the IP layer. (A companion document, [I-D.ietf-conex-abstract-mech], focuses on the mechanics of the protocol.) Briefly, the idea is for the sender to continually signal expected congestion in the headers of any data it sends. To a first approximation, the sender does this by relaying the 'Congestion Feedback Signals' back into the IP layer. They then travel unchanged across the network to the receiver (shown as 'IP-Layer-ConEx-Signals' in Figure 1). This enables IP layer devices on the path to see information about the whole path congestion.

```
,---------.                                               ,---------.
|Transport|                                               |Transport|
| Sender  |   .                                           |Receiver |
|         |  /|___________________________________________|         |
|     ,-<---------------Congestion-Feedback-Signals--<--------.     |
|     |   |/                                              |   |     |
|     |   |\           Transport Layer Feedback Flow      |   |     |
|     |   | \  ___________________________________________|   |     |
|     |   |  \|                                           |   |     |
|     |   |   '         ,-----------.               .     |   |     |
|     |   |_____________|           |_______________|\    |   |     |
|     |   |    IP Layer |           |  Data Flow      \   |   |     |
|     |   |             |(Congested)|                  \  |   |     |
|     |   |             |  Network  |--Congestion-Signals--->-'     |
|     |   |             |  Device   |                    \|         |
|     |   |             |           |                    /|         |
|     `----------->--(new)-IP-Layer-ConEx-Signals-------->|         |
|         |             |           |                  /  |         |
|         |_____________|           |_______________  /   |         |
|         |             |           |               |/    |         |
`---------'             `-----------'               '     `---------'
```

Figure 1: The ConEx Protocol in the Internet Architecture

One of the key benefits of exposing this congestion information at the IP layer is that it makes the information available to network operators for use as input into their traffic management procedures. A ConEx-enabled sender signals expected whole path congestion, which is approximately the congestion at least a round trip time earlier as reported by the receiver to the sender (Figure 1). The ConEx signal is a mark in the IP header that is easy for any IP device to read. Therefore a node performing traffic management can count congestion as easily as it might count data volume today by simply counting the volume of packets with ConEx markings.

ConEx-based traffic management can make highly efficient use of capacity. In times of no congestion, all traffic management restraints can be removed, leaving the network's full capacity available to all its users. If some users on the network cause disproportionate congestion, the traffic management function can learn about this and directly limit those users' traffic in order to protect the service of other users sharing the same capacity. ConEx-based traffic management thus presents a step change in terms of the options available to network operators for managing traffic on their networks.

The remainder of this document explains the concepts behind ConEx and how exposing congestion can significantly improve Internet traffic management, among other benefits. Section 2 introduces a number of concepts that are fundamental to understanding how ConEx-based traffic management works. Section 3 shows how ConEx can be used for traffic management, discusses additional benefits from such usage, and compares ConEx-based traffic management to existing traffic management approaches. Section 4 discusses other related use cases. Section 5 briefly discusses deployment arrangements. The final sections are standard RFC back matter.

The remainder of the core ConEx document suite consists of:

[I-D.ietf-conex-abstract-mech], which provides an abstract encoding of ConEx signals, explains the ConEx audit and security mechanisms, and describes incremental deployment features;

[I-D.ietf-conex-destopt], which specifies the IPv6 destination option encoding for ConEx;

[I-D.ietf-conex-tcp-modifications], which specifies TCP sender modifications for use of ConEx;

and the following documents, which describe some feasible scenarios for deploying ConEx:

[I-D.briscoe-conex-initial-deploy], which describes a scenario around a fixed broadband access network;

[I-D.ietf-conex-mobile], which describes a scenario around a mobile communications provider;

[I-D.briscoe-conex-data-centre], which describes how ConEx could be used for performance isolation between tenants of a data centre.

2. Concepts

ConEx relies on a precise definition of congestion and a number of newer concepts that are introduced in this section. Definitions are summarized in Section 2.4.

2.1. Congestion

Despite its central role in network control and management, congestion is a remarkably difficult concept to define. Experts in different disciplines and with different perspectives define congestion in a variety of ways [Bauer09].

The definition used for the purposes of ConEx is expressed as the probability of packet loss (or the probability of packet marking if ECN is in use). This definition focuses on how congestion is measured, rather than describing congestion as a condition or state.

2.2. Congestion-Volume

The metric that ConEx exposes is congestion-volume: the volume of bytes dropped or ECN-marked in a given period of time. Counting congestion-volume allows each user to be held responsible for his or her contribution to congestion. Congestion-volume can only be a property of traffic, whereas congestion can be a property of traffic or a property of a link or a path.

To understand congestion-volume, consider a simple example. Imagine Alice sends 1GB of a file while the loss-probability is a constant 0.2%. Her contribution to congestion -- her congestion-volume -- is 1GB x 0.2% = 2MB. If she then sends another 3GB of the file while the loss-probability is 0.1%, this adds 3MB to her congestion-volume. Her total contribution to congestion is then 2MB+3MB = 5MB.

Fortunately, measuring Alice’s congestion-volume on a real network does not require the kind of arithmetic shown above because congestion-volume can be directly measured by counting the total volume of Alice’s traffic that gets discarded or ECN-marked. (A queue with varying percentage loss does these multiplications and additions inherently.) With ConEx, network operators can count congestion-volume using techniques very similar to those they use for counting volume.

2.3. Rest-of-Path Congestion

At a particular measurement point within a network, "rest-of-path congestion" (also known as "downstream congestion") is the level of congestion that a traffic flow is expected to experience between the measurement point and its final destination. "Upstream congestion" is the congestion experienced up to the measurement point.

If traffic is ECN-capable, ECN signals monitored in the middle of a network will indicate the congestion experienced so far on the path (upstream congestion). In contrast, the ConEx signals inserted into IP headers as shown in Figure 1 indicate the congestion along a whole path from transport source to transport destination. Therefore if a measurement point detects both of these signals, it can subtract the level of ECN (upstream congestion) from the level of ConEx (whole path) to derive a measure of the congestion that packets are likely to experience between the monitoring point and their destination (rest-of-path congestion). A measurement point can calculate this measurement in the aggregate, across all flows.

A network monitor can usually accurately measure upstream congestion only if the traffic it observes is ECN-capable. [I-D.ietf-conex-abstract-mech] has further discussion of the constraints around the network’s ability to measure upstream and rest-of-path congestion in these circumstances. However, there are a number of intial deployment arrangements that benefit from ConEx but work without ECN (see Section 5).

2.4. Definitions

Congestion: In general, congestion occurs when any user’s traffic suffers loss, ECN marking, or increased delay as a result of one or more network resources becoming overloaded. For the purposes of ConEx, congestion is measured using the concrete signals provided by loss and ECN markings (delay is not considered). Congestion is measured as the probability of loss or the probability of ECN marking, usually expressed as a dimensionless percentage.

Congestion-volume: For any granularity of traffic (packet, flow, aggregate, link, etc.), the volume of bytes dropped or ECN-marked in a given period of time. Conceptually, data volume multiplied by the congestion each packet of the volume experienced. Usually expressed in bytes (or MB or GB).

Congestion policer: A logical entity that allows a network operator to monitor each user's congestion-volume and enforce congestion-volume limits (discussed in Section 3.1).

Rest-of-path congestion (or downstream congestion): The congestion a flow of traffic is expected to experience on the remainder of its path. In other words, at a measurement point in the network, the rest-of-path congestion is the congestion the traffic flow has yet to experience as it travels from that point to the receiver. This is usually expressed as a dimensionless percentage.

Upstream congestion: The accumulated congestion experienced by a traffic flow thus far, relative to a point along its path. In other words, at a measurement point in the network the upstream congestion is the accumulated congestion the traffic flow has experienced as it travels from the sender to that point. At the receiver this is equivalent to the end-to-end congestion level that (usually) is reported back to the sender. This is usually expressed as a dimensionless percentage.

Network operators (or providers): Operator of a residential, commercial, enterprise, campus or other network.

User: The contractual entity that represents an individual, household, business, or institution that uses the service of a network operator. There is no implication that the contract has to be commercial; for instance, the users of a university or enterprise network service could be students or employees who do not pay for access but may be required to comply with some form of contract or acceptable use policy. There is also no implication that every user is an end user. Where two networks form a customer-provider relationship, the term user applies to the customer network.

[I-D.ietf-conex-abstract-mech] gives further definitions for aspects of ConEx related to protocol mechanisms.

3. Core Use Case: Informing Traffic Management

This section explains how ConEx could be used as the basis for traffic management, highlights additional benefits derived from having ConEx-aware nodes on the network, and compares ConEx-based traffic management to existing approaches.

3.1. Use Case Description

One of the key benefits that ConEx can deliver is in helping network operators to improve how they manage traffic on their networks. Consider the common case of a commercial broadband network where a relatively small number of users place disproportionate demand on network resources, at times resulting in congestion. The network operator seeks a way to manage traffic such that the traffic that contributes more to congestion bears more of the brunt of the management.

Assuming ConEx signals are visible at the IP layer, the network operator can accomplish this by placing a congestion policer at an enforcement point within the network and configuring it with a traffic management policy that monitors each user’s contribution to congestion. As described in [I-D.ietf-conex-abstract-mech] and elaborated in [CongPol], one way to implement a congestion policer is in a similar way to a bit-rate policer, except that it monitors congestion-volume (based on IP layer ConEx signals) rather than bit-rate. When implemented as a token bucket, the tokens provide users with the right to cause bits of congestion-volume, rather than to send bits of data volume. The fill rate represents each user’s congestion-volume quota.

The congestion policer monitors the ConEx signals of the traffic entering the network. As long as the network remains uncongested and users stay within their quotas, no action is taken. When the network becomes congested and a user exhausts his quota, some action is taken against the traffic that breached the quota in accordance with the network operator’s traffic management policy. For example, the traffic may be dropped, delayed, or marked with a lower QoS class. In this way, traffic is managed according to its contribution to congestion -- not some application- or flow-specific policy -- and is not managed at all during times of no congestion.

As an example of how a network operator might employ a ConEx-based traffic management system, consider a typical DSL network architecture (as elaborated in [TR-059] and [TR-101]). Traffic is routed from regional and global IP networks to an operator-controlled IP node, the Broadband Remote Access Server (BRAS). From the BRAS, traffic is delivered to access nodes. The BRAS carries enhanced functionality including IP QoS and traffic management capabilities.

By deploying a congestion policer at the BRAS location, the network operator can measure the congestion-volume created by users within the access nodes and police misbehaving users before their traffic affects others on the access network. The policer would be provisioned with a traffic management policy, perhaps directing the BRAS to drop packets from users that exceed their congestion-volume quotas during times of congestion. Those users' apps would be likely to react in the typical way to drops, backing off (assuming at least some use TCP), and thereby lowering the users' congestion-volumes back within the quota limits. If none of a user's apps responds, the policer would continue to increase focused drops and effectively enforce its own congestion control.

3.2. Additional Benefits

The ConEx-based approach to traffic management has a number of benefits in addition to efficient management of traffic. It provides incentives for users to make use of "scavenger" transport protocols, such as [I-D.ietf-ledbat-congestion], that provide ways for bulk-transfer applications to rapidly yield when interactive applications require capacity (thereby "scavenging" remaining bandwidth). With a congestion policer in place as described in Section 3.1, users of these protocols will be less likely to run afoul of the network operator's traffic management policy than those whose bulk-transfer applications generate the same volume of traffic without being sensitive to congestion. In short, two users who produce similar traffic volumes over the same time interval may produce different congestion-volumes if one of them is using a scavenger transport protocol and the other is not; in that situation the scavenger user's traffic is less likely to be managed by the network operator.

ConEx-based traffic management also makes it possible for a user to control the relative performance among its own traffic flows. If a user wants some flows to have more bandwidth than others, it can reduce the rate of some traffic so that it consumes less congestion-volume "budget", leaving more congestion-volume "budget" for the user to "spend" on making other traffic go faster. This approach is most relevant if congestion is signalled by ECN, because no impairment due to loss is involved and delay can remain low.

3.3. Comparison with Existing Approaches

A variety of approaches already exist for network operators to manage congestion, traffic, and the disproportionate usage of scarce capacity by a small number of users. Common approaches can be categorized as rate-based, volume-based, or application-based.

Rate-based approaches constrain the traffic rate per user or per network. A user's peak and average (or "committed") rate may be limited. These approaches have the potential to either over- or under-constrain the network, suppressing rates even when the network is uncongested or not suppressing them enough during heavy usage periods.

Round-robin scheduling and fair queuing were developed to address these problems. They equalize relative rates between active users (or flows) at a known bottleneck. The bit-rate allocated to any one user depends on the number of active users at each instant. The drawback of these approaches is that they favor heavy users over light users over time, because they do not have any memory of usage. Heavy users will be active at every instant whereas light users will only occupy their share of the link occassionally, but bit-rate is shared instant by instant.

Volume-based approaches measure the overall volume of traffic a user sends (and/or receives) over time. Users may be subject to an absolute volume cap (for example, 10GB per month) or the "heaviest" users may be sanctioned in some other manner. Many providers use monthly volume limits and count volume regardless of whether the network is congested or not, creating the potential for over- or under-constraining problems, as with the original rate-based approaches.

ConEx-based approaches, by comparison, only react during times of congestion and in proportion to each user's congestion contribution, making more efficient use of capacity and more proportionate management decisions.

Unlike ConEx-based approaches, neither rate-based nor volume-based approaches provide incentives for applications to use scavenger transport protocols. They may even penalize users of applications that employ scavenger transports for the large amount of volume they send, rather than rewarding them for carefully avoiding congestion while sending it. While the volume-based approach described in Comcast's Protocol-Agnostic Congestion Management System [RFC6057] aims to overcome the over/under-constraining problem by only measuring volume and triggering traffic management action during periods of high utilization, it still does not provide incentives to use scavenger transports because congestion-causing volume cannot be distinguished from volume overall. ConEx provides this ability.

Application-based approaches use deep packet inspection or other techniques to determine what application a given traffic flow is associated with. Network operators may then use this information to rate-limit or otherwise sanction certain applications, in some cases only during peak hours. These approaches suffer from being at odds with IPsec and some application-layer encryption, and they may raise additional policy concerns. In contrast, ConEx offers an application-agnostic metric to serve as the basis for traffic management decisions.

The existing types of approaches share a further limitation that ConEx can help to overcome: performance uncertainty. Flat-rate pricing plans are popular because users appreciate the certainty of having their monthly bill amount remain the same for each billing period, allowing them to plan their costs accordingly. But while flat-rate pricing avoids billing uncertainty, it creates performance uncertainty: users cannot know whether the performance of their connections is being altered or degraded based on how the network operator is attempting to manage congestion. By exposing congestion information at the IP layer, ConEx instead provides a metric that can serve as an open, transparent basis for traffic management policies that both providers and their customers can measure and verify. It can be used to reduce the performance uncertainty that some users currently experience.

4. Other Use Cases

ConEx information can be put to a number of uses other than informing traffic management. These include:

Informing inter-operator contracts: ConEx information is made visible to every IP node, including border nodes between networks. Network operators can use ConEx combined with ECN markings to measure how much traffic from each network contributes to congestion in the other. As such, congestion-volume could be included as a metric in inter-operator contracts, just as volume or bit-rate are included today. This would not be an initial deployment scenario, unless ECN became widely deployed.

Enabling more efficient capacity provisioning: Section 3.2 explained how operators can use ConEx-based traffic management to encourage use of scavenger transport protocols, which significantly improves the performance of interactive applications while still allowing heavy users to transfer high volumes. Here we explain how this can also benefit network operators.

Today, when loss, delay or averaged utilization exceeds a certain threshold, some operators just buy more capacity without attempting to manage the traffic. Other operators prefer to limit a minority of heavy users at peak times, but they still eventually buy more capacity when utilization rises.

With ConEx-based traffic management, a network operator should be able to provision capacity more efficiently. An operator could benefit from this in a variety of ways. For example, the operator could add capacity as it would do without ConEx, but deliver better quality of service for its users. Or the operator could delay adding capacity while delivering similar quality of service to what it currently provides.

5. Deployment Arrangements

ConEx is designed so that it can be incrementally deployed in the Internet and still be valuable for early adopters. As long as some senders are ConEx-enabled, a network on the path can unilaterally use ConEx-aware policy devices for traffic management; no changes to network forwarding elements are needed and ConEx still works if there are other networks on the path that are unaware of ConEx marks.

The above two steps seem to represent a stand-off where neither step is useful until the other has made the first move: i) some sending hosts must be modifed to give information to the network and ii) a network must deploy policy devices to monitor this information and act on it. Nonetheless, the developer of a scavenger transport protocol like LEDBAT does stand to benefit from deploying ConEx. In this case the developer makes the first move, expecting it will prompt at least some networks to move in response, using the ConEx information to reward users of the scavenger transport protocol.

On the host side, we have already shown (Figure 1) how the sender piggy-backs ConEx signals on normal data packets to re-insert feedback about packet drops (and/or ECN) back into the IP layer. In the case of TCP, [I-D.ietf-conex-tcp-modifications] proposes the required sender modifications. ConEx works with any TCP receiver as long as it uses SACK, which most do. There is a receiver optimisation [I-D.tcpm-accurate-ecn] that improves ConEx precision when using ECN, but ConEx can still use ECN without it. Networks can make use of ConEx even if the implementations of some of the transport protocols on a host do not support ConEx (e.g. the implementation of DNS over UDP might not support ConEx, while perhaps RTP over UDP and TCP will).

On the network side the provider solely needs to place ConEx congestion policers at each ingress to its network, in a similar arrangement to the edge-policed architecture of Diffserv [RFC2475].

A sender can choose whether to send packets that support ConEx or packets that don’t. ConEx-enabled packets bring information to the policer about congestion expected on the rest of the path beyond the policer. Packets that do not support ConEx bring no such information. Therefore the network will tend to conservatively rate-limit non-ConEx-enabled packets in order to manage the unknown risk of congestion. In contrast, a network doesn’t normally need to rate-limit ConEx-enabled packets unless they reveal a persistently high contribution to congestion. This natural tendency for networks to favour senders that provide ConEx information reinforces ConEx deployment.

Feasible initial deployment scenarios exist for a broadband access network [I-D.briscoe-conex-initial-deploy], a mobile communications network [I-D.ietf-conex-mobile], and a multi-tenant data centre [I-D.briscoe-conex-data-centre]. The first two of these scenarios are believed to work well without ECN support, while the data center scenario works best with ECN (where it may be more likely for ECN to be deployed in the future).

The above gives only the most salient aspects of ConEx deployment. For further detail, [I-D.ietf-conex-abstract-mech] describes the incremental deployment features of the ConEx protocol and the components that need to be deployed for ConEx to work.

6. Experimental Considerations

ConEx is initially designed as an experimental protocol because it makes an ambitious change at the interoperability (IP) layer, so no amount of careful design can foresee all the potential feature interactions with other uses of IP. This section identifies a number of questions that would be useful to answer through well-designed experiments:

- Are the compromises that were made in order to fit the ConEx encoding into IP (for example, that the initial design was solely for IPv6 and not for IPv4, and that the encoding has limited visibility when tunnelled [I-D.ietf-conex-destopt]) the right ones?

- Is it possible to combine techniques for distinguishing self-congestion from shared congestion with ConEx-based traffic management such that users are not penalized for congestion that does not impact others on the network? Are other techniques needed?

- If ECN deployment remains patchy, are the proposed initial ConEx deployment scenarios (Section 5) still useful enough to kick-start deployment? Is audit effective when based on loss at a primary bottleneck? Can rest-of-path congestion be approximated accurately enough without ECN? Are there other useful deployment scenarios?

- In practice, how does traffic management using ConEx compare with traditional techniques (Section 3.3)? Does it give the benefits claimed in Section 3.1 and Section 3.2?

- Approaches are proposed for congestion policing of ConEx traffic alongside existing management (or lack thereof) of non-ConEx traffic, including UDP traffic [I-D.ietf-conex-abstract-mech]. Are they strategy-proof against users selectively using both? Are there better transition strategies?

- Audit devices have been designed and implemented to assure ConEx signal integrity [I-D.ietf-conex-abstract-mech]. Do they achieve minimal false hits and false misses in a wide range of traffic scenarios? Are there new attacks? Are there better audit designs to defend against these?

ConEx is intended to be a generative technology that might be used for unexpected purposes unforeseen by the designers. Therefore this list of experimental considerations is not intended to be exhaustive.

7. Security Considerations

This document does not specify a mechanism, it merely motivates congestion exposure at the IP layer. Therefore security considerations are described in the companion document that gives an abstract description of the ConEx protocol and the components that would use it [I-D.ietf-conex-abstract-mech].

8. IANA Considerations

This document does not require actions by IANA.

9. Acknowledgments

Bob Briscoe was partly funded by Trilogy, a research project (ICT-216372) supported by the European Community under its Seventh Framework Programme. The views expressed here are those of the author only.

The authors would like to thank the many people that have commented on this document: Bernard Aboba, Mikael Abrahamsson, Joao Taveira Araujo, Marcelo Bagnulo Braun, Steve Bauer, Caitlin Bestler, Steven Blake, Louise Burness, Ken Carlberg, Nandita Dukkipati, Dave McDysan, Wes Eddy, Matthew Ford, Ingemar Johansson, Georgios Karagiannis, Mirja Kuehlewind, Dirk Kutscher, Zhu Lei, Kevin Mason, Matt Mathis, Michael Menth, Chris Morrow, Tim Shepard, Hannes Tschofenig and Stuart Venters. Please accept our apologies if your name has been missed off this list.

9.1. Contributors

Philip Eardley and Andrea Soppera made helpful text contributions to this document.

The following co-edited this document through most of its life:

Toby Moncaster
Computer Laboratory
William Gates Building
JJ Thomson Avenue
Cambridge, CB3 0FD
UK
EMail: toby.moncaster@cl.cam.ac.uk

John Leslie
JLC.net
10 Souhegan Street
Milford, NH 03055
US
EMail: john@jlc.net

10. Informative References

[Bauer09] Bauer, S., Clark, D., and W. Lehr, "The Evolution of Internet Congestion", 2009.

[CongPol] Briscoe, B., Jacquet, A., and T. Moncaster, "Policing Freedom to Use the Internet Resource Pool", RE-Arch 2008 hosted at the 2008 CoNEXT conference , December 2008.

[I-D.briscoe-conex-data-centre] Briscoe, B. and M. Sridharan, "Network Performance Isolation in Data Centres using Congestion Exposure (ConEx)", draft-briscoe-conex-data-centre-00 (work in progress), July 2012.

[I-D.briscoe-conex-initial-deploy] Briscoe, B., "Initial Congestion Exposure (ConEx) Deployment Examples", draft-briscoe-conex-initial-deploy-02 (work in progress), March 2012.

[I-D.ietf-conex-abstract-mech] Mathis, M. and B. Briscoe, "Congestion Exposure (ConEx) Concepts and Abstract Mechanism", draft-ietf-conex-abstract-mech-04 (work in progress), March 2012.

[I-D.ietf-conex-destopt] Krishnan, S., Kuehlewind, M., and C. Ucendo, "IPv6 Destination Option for Conex", draft-ietf-conex-destopt-02 (work in progress), March 2012.

[I-D.ietf-conex-mobile] Kutscher, D., Mir, F., Winter, R., Krishnan, S., Zhang, Y., and C. Bernardos, "Mobile Communication Congestion Exposure Scenario", draft-ietf-conex-mobile-00 (work in progress), July 2012.

[I-D.ietf-conex-tcp-modifications] Kuehlewind, M. and R. Scheffenegger, "TCP modifications for Congestion Exposure", draft-ietf-conex-tcp-modifications-02 (work in progress), May 2012.

[I-D.ietf-ledbat-congestion] Hazel, G., Iyengar, J., Kuehlewind, M., and S. Shalunov, "Low Extra Delay Background Transport (LEDBAT)", draft-ietf-ledbat-congestion-09 (work in progress), October 2011.

[I-D.tcpm-accurate-ecn] Kuehlewind, M. and R. Scheffenegger, "Accurate ECN Feedback Option in TCP", draft-kuehlewind-tcpm-accurate-ecn-option-01 (work in progress), July 2012.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of

Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC6057] Bastian, C., Klieber, T., Livingood, J., Mills, J., and R. Woundy, "Comcast's Protocol-Agnostic Congestion Management System", RFC 6057, December 2010.

[TR-059] Anschutz, T., Ed., "DSL Forum Technical Report TR-059: Requirements for the Support of QoS-Enabled IP Services", September 2003.

[TR-101] Cohen, A., Ed. and E. Schrum, Ed., "DSL Forum Technical Report TR-101: Migration to Ethernet-Based DSL Aggregation", April 2006.

Authors' Addresses

Bob Briscoe (editor)
BT
B54/77, Adastral Park
Martlesham Heath
Ipswich IP5 3RE
UK

Phone: +44 1473 645196
EMail: bob.briscoe@bt.com
URI: http://bobbriscoe.net/

Richard Woundy (editor)
Comcast
1701 John F Kennedy Boulevard
Philadelphia, PA 19103
US

EMail: richard_woundy@cable.comcast.com
URI: http://www.comcast.com

Alissa Cooper (editor)
CDT
1634 Eye St. NW, Suite 1100
Washington, DC 20006
US

EMail: acooper@cdt.org

S. Krishnan
Ericsson
M. Kuehlewind
IKR University of Stuttgart
C. Ucendo
Telefonica

July 4, 2011

IPv6 Destination Option for Conex
draft-krishnan-conex-destopt-00

Abstract

Conex is a mechanism by which senders inform the network about the congestion encountered by packets earlier in the same flow. This document specifies an IPv6 destination option that is capable of carrying conex markings in IPv6 datagrams.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction 3
2. Conventions used in this document 3
3. Background 3
4. Conex Destination Option (CDO) 4
5. Acknowledgements 5
6. Security Considerations 5
7. IANA Considerations 5
8. Normative References 5
Authors' Addresses 5

1. Introduction

Conex is a mechanism by which senders inform the network about the congestion encountered by packets earlier in the same flow. This document specifies an IPv6 destination option that can be used for performing conex markings in IPv6 datagrams.

2. Conventions used in this document

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL","SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Background

The Conex working group came up with a list of requirements that had to be met by any marking mechanism. It then considered several alternative mechanisms and evaluated their suitability for conex marking. There were no mechanisms found that were completely suitable, but the only mechanism that came close to meeting the requirements was IPv6 destination options. The analysis of the different alternatives can be found in [draft-krishnan-conex-ipv6].

4. Conex Destination Option (CDO)

The Conex Destination Option (CDO) is a destination option that can be included in IPv6 datagrams that are sent by conex-aware senders in order to inform conex-aware nodes on the path about the CDO has an alignment requirement of (none).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
                                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                                |  Option Type  | Option Length |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|X|L|E|C|                       Reserved                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 1: Conex Destination Option Layout

Option Type

8-bit identifier of the type of option. The option identifier for the conex destination option will be allocated by the IANA.

Option Length

8-bit unsigned integer. The length of the option (excluding the Option Type and Option Length fields). This field MUST be set to the value 4.

X Bit
When this bit is set, the transport sender is using ConEx with this packet. If it is reset, the sender is not using ConEx.

L Bit

When this bit is set, the transport sender has experienced a loss. If it is reset, the sender has not experienced a loss.

E Bit

When this bit is set, the transport sender has experienced ECN-signaled congestion. If it is reset, the sender has not experienced ECN-signaled congestion.

C Bit

When this bit is set, the transport sender is building up congestion credit. Otherwise it is not.

5. Acknowledgements

The authors would like to thank Marcelo Bagnulo, Bob Briscoe, Ingemar Johansson, Joel Halpern and John Leslie for the discussions that led to this document.

6. Security Considerations

This document does not bring up any new security issues.

7. IANA Considerations

This document defines a new IPv6 destination option for carrying conex markings. IANA is requested to assign a new destination option type in the Destination Options registry maintained at http://www.iana.org/assignments/ipv6-parameters <TBA1> Conex Destination Option [RFCXXXX] The act bits for this option need to be 10 and the chg bit needs to be 0.

8. Normative References

[CAM] Briscoe, B., "Congestion Exposure (ConEx) Concepts and Abstract Mechanism", draft-ietf-conex-abstract-mech-01 (work in progress), March 2011.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

Authors' Addresses

Suresh Krishnan
Ericsson
8400 Blvd Decarie
Town of Mount Royal, Quebec
Canada

Email: suresh.krishnan@ericsson.com

Mirja Kuehlewind
IKR University of Stuttgart

Email: mirja.kuehlewind@ikr.uni-stuttgart.de

Carlos Ralli Ucendo
Telefonica

Email: ralli@tid.es

Congestion Exposure (ConEx) M. Kuehlewind, Ed.
Internet-Draft University of Stuttgart
Intended status: Experimental R. Scheffenegger
Expires: May 3, 2012 NetApp, Inc.
October 31, 2011

Accurate ECN Feedback in TCP
draft-kuehlewind-conex-accurate-ecn-01

Abstract

Explicit Congestion Notification (ECN) is an IP/TCP mechanism where network nodes can mark IP packets instead of dropping them to indicate congestion to the end-points. An ECN-capable receiver will feedback this information to the sender. ECN is specified for TCP in such a way that only one feedback signal can be transmitted per Round-Trip Time (RTT). Recently new TCP mechanisms like ConEx or DCTCP need more accurate feedback information in the case where more than one marking is received in one RTT.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction 3
  1.1. Overview ECN and ECN Nonce in TCP 4
  1.2. Design choices 4
  1.3. Requirements Language 5
2. Negotiation in TCP handshake 6
3. Accurate Feedback 7
  3.1. Coding 7
    3.1.1. Requirements 7
    3.1.2. One bit feedback flag 9
      3.1.2.1. Discussion 10
    3.1.3. Three bit field with counter feedback 11
      3.1.3.1. Discussion 12
    3.1.4. Codepoints with dual counter feedback 13
      3.1.4.1. Implementation 14
      3.1.4.2. Discussion 15
    3.1.5. Short Summary of the Discussions 16
  3.2. TCP Sender 17
  3.3. TCP Receiver 17
  3.4. Advanced Compatibility Mode 17
4. Acknowledgements 18
5. IANA Considerations 18
6. Security Considerations 18
7. References 19
  7.1. Normative References 19
  7.2. Informative References 19
Appendix A. Pseudo Code for the Codepoint Coding 19
Authors' Addresses 22

1. Introduction

Explicit Congestion Notification (ECN) [RFC3168] is an IP/TCP mechanism where network nodes can mark IP packets instead of dropping them to indicate congestion to the end-points. An ECN-capable receiver will feedback this information to the sender. ECN is specified for TCP in such a way that only one feedback signal can be transmitted per Round-Trip Time (RTT). Recently proposed mechanisms like Congestion Exposure (ConEx) or DCTCP [Ali10] need more accurate feedback information in case when more than one marking is received in one RTT.

This documents discusses and (will in a further version specify) a different scheme for the ECN feedback in the TCP header to provide more than one feedback signal per RTT. This modification does not obsolete [RFC3168]. It provides an extension that requires additional negotiation in the TCP handshake by using the TCP nonce sum (NS) bit which is currently not used when SYN is set.

In the current version of this document there are different coding schemes proposed for discussion. All proposed codings aim to scope with the given bit space. All schemes require the use of the NS bit at least in the TCP handshake. Depending of the coding scheme the accurate ECN feedback extension will or will not include the ECN-Nonce integrity mechanism. A later version of this document will choose between the coding options, and remove the rationale for the choice and the specs of those schemes not chosen. If a scheme will be chosen that does not include ECN Nonce, a mechanism that is requiring a more accurate ECN feedback needs to provide an own method to ensure the integrity of the congestion feedback information or has to scope with the uncertainty of this information.

The following scenarios should briefly show where the accurate feedback is needed or provides additional value:

a. A Standard TCP sender with [RFC5681] congestion control algorithm that supports ConEx:
   In this case the congestion control algorithm still ignores multiple marks per RTT, while the ConEx mechanism uses the extra information per RTT to re-echo more precise congestion information.

b. A sender using DCTCP without ConEx:
   The congestion control algorithm uses the extra info per RTT to perform its decrease depending on the number of congestion marks.

c. A sender using DCTCP congestion control and supports ConEx:
   Both the congestion control algorithm and ConEx use the accurate

ECN feedback mechanism.

d. A standard TCP sender using RFC5681 congestion control algorithm without ConEx:
No accurate feedback is necessary here. The congestion control algorithm still react only on one signal per RTT. But its best to have one generic feedback mechanism, whether you use it or not.

1.1. Overview ECN and ECN Nonce in TCP

ECN requires two bits in the IP header. The ECN capability of a packet is indicated, when either one of the two bits is set. An ECN sender can set one or the other bit to indicate an ECN-capable transport (ETC) which results in two signals --- ECT(0) and respectively ECT(1). A network node can set both bits simultaneously when it experiences congestion. When both bits are set the packets is regarded as "Congestion Experienced" (CE).

In the TCP header two bits in byte 14 are defined for the use of ECN. The TCP mechanism for signaling the reception of a congestion mark uses the ECN-Echo (ECE) flag in the TCP header. To enable the TCP receiver to determine when to stop setting the ECN-Echo flag, the CWR flag is set by the sender upon reception of the feedback signal.

ECN-Nonce [RFC3540] is an optional addition to ECN that is used to protects the TCP sender against accidental or malicious concealment of marked or dropped packets. This addition defines the last bit of the 13 byte in the TCP header as the Nonce Sum (NS) bit. With ECN-Nonce a nonce sum is maintain that counts the occurrence of ECT(1) packets.

```
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               |           | N | C | E | U | A | P | R | S | F |
| Header Length | Reserved  | S | W | C | R | C | S | S | Y | I |
|               |           |   | R | E | G | K | H | T | N | N |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Figure 1: The (post-ECN Nonce) definition of the TCP header flags

1.2. Design choices

The idea of this document is to use the ECE, CWR and NS bits for additional capability negotiation during the SYN/SYN-ACK exchange, and then for the more accurate feedback itself on subsequent packets in the flow (with SYN=0).

Alternatively, a new TCP option could be introduced, to help maintain the accuracy, and integrity of the ECN feedback between receiver and sender. Such an option could provide more information. E.g. ECN for RTP/UDP provides explicit the number of ECT(0), ECT(1), CE, non-ECT marked and lost packets. However, deploying new TCP options has it’s own challenges. A seperate documents proposed a new TCP Option for accurate ECN feedback. This option could be used in addition to an more accurate ECN feedback scheme described here or in addtion to the classic ECN, when available and needed.

As seen in Figure 1, there are currently three unused flag bits in the TCP header. Any of the below described schemes could be extended by one or more bits, to add higher resiliency against ACK loss. The relative gains would be proportional to each of the described schemes, while the respective drawbacks would remain identical. Thus the approach in this document is to scope with the given number of bits as they seem to be already sufficient and the accurate ECN feedback scheme will only be used instead of the classic ECN and never in parallel.

1.3. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

We use the following terminology from [RFC3168] and [RFC3540]:

The ECN field in the IP header:

CE: the Congestion Experienced codepoint; and

ECT(0)/ECT(1): either one of the two ECN-Capable Transport codepoints.

The ECN flags in the TCP header:

CWR: the Congestion Window Reduced flag;

ECE: the ECN-Echo flag; and

NS: ECN Nonce Sum.

In this document, we will call the ECN feedback scheme as specified in [RFC3168] the ’classic ECN’ and our new proposal the ’accurate ECN feedback’ scheme. A ’congestion mark’ is defined as an IP packet where the CE codepoint is set. A ’congestion event’ refers to one or more congestion marks belong to the same overload situation in the network (usually during one RTT).

2. Negotiation in TCP handshake

During the TCP hand-shake at the start of a connection, an originator of the connection (host A) MUST indicate a request to get more accurate ECN feedback by setting the TCP flags NS=1, CWR=1 and ECE=1 in the initial SYN.

A responding host (host B) MUST return a SYN ACK with flags CWR=1 and ECE=0. The responding host MUST NOT set this combination of flags unless the preceding SYN has already requested support for accurate ECN feedback as above. Normally a server (B) will reply to a client with NS=0, but if the initial SYN from client A is marked CE, the sever B can set the NS flag to 1 to indicate the congestion immediately instead of delaying the signal to the first acknowledgment when the actually data transmission already started. So, server B MAY set the alternative TCP header flags in its SYN ACK: NS=1, CWR=1 and ECE=0.

The Addition of ECN to TCP SYN/ACK packets is discussed and specified as experimental in [RFC5562]. The addition of ECN to the SYN packet is optional. The security implication when using this option are not further discussed here.

These handshakes are summarized in Table 1 below, with X indicating NS can be either 0 or 1 depending on whether congestion had been experienced. The handshakes used for the other flavors of ECN are also shown for comparison. To compress the width of the table, the headings of the first four columns have been severely abbreviated, as follows:

Ac: *Ac*curate ECN Feedback

N: ECN-*N*once (RFC3540)

E: *E*CN (RFC3168)

I: Not-ECN (*I*mplicit congestion notification).

Ac	N	E	I	[SYN] A->B	[SYN,ACK] B->A	Mode
				NS CWR ECE	NS CWR ECE	
AB				1 1 1	X 1 0	accurate ECN
A	B			1 1 1	1 0 1	ECN Nonce
A		B		1 1 1	0 0 1	classic ECN
A			B	1 1 1	0 0 0	Not ECN
A			B	1 1 1	1 1 1	Not ECN (broken)

Table 1: ECN capability negotiation between Sender (A) and Receiver (B)

Recall that, if the SYN ACK reflects the same flag settings as the preceding SYN (because there is a broken RFC3168 compliant implementation that behaves this way), RFC3168 specifies that the whole connection MUST revert to Not-ECT.

3. Accurate Feedback

In this section we refer the sender to be the on sending data and the receiver as the one that will acknowledge this data. Of course such a scenario is describing only one half connection of a TCP connection. The proposed scheme, if negotiated, will be used for both half connection as both, sender and receiver, need to be capable to echo and understand the accurate ECN feedback scheme.

3.1. Coding

This section proposes three different coding schemes for discussion. First, requirements are listed that will allow to evaluate the proposed schemes against each other. A later version of this document will choose between the coding options, and remove the rationale for the choice and the specs of those schemes not chosen. The next section provides basically a fourth alternative to allow a compatibility mode when a sender needs accurate feedback but has to operate with a legacy [RFC3168] receiver.

3.1.1. Requirements

The requirements of the accurate ECN feedback protocol for the use of e.g. Conex or DCTCP are to have a fairly accurate (not necessarily perfect), timely and protected signaling. This leads to the following requirements:

Resilience

The ECN feedback signal is implicit carried within the TCP acknowledgment. TCP ACKs can get lost. Moreover, delayed ACK are usually used with TCP. That means in most cases only every second data packets gets acknowledged. In a high congestion situation where most of the packet are marked with CE, an accurate feedback mechanism must still be able to signal sufficient congestion information. Thus the accurate ECN feedback extension has to take delayed ACK and ACK loss into account.

Timely

The CE marking is induced by a network node on the transmission path and echoed by the receiver in the TCP acknowledgment. Thus when this information arrives at the sender, its naturally already about one RTT old. With a sufficient ACK rate a further delay of a small number of ACK can be tolerated but with large delays this information will be out dated due to high dynamic in the network. TCP congestion control which introduces parts of this dynamic operates on an time scale of one RTT. Thus the congestion feedback information should be delivered timely (within one RTT).

Integrity

With ECN Nonce, a misbehaving receiver can be detected with a certain probability. As this accurate ECN feedback might reuse the NS bit it is encouraged to ensure integrity as least as good as ECN Nonce. If this is not possible, alternative approaches should be provided how a mechanism using the accurate ECN feedback extension can re-ensure integrity or give strong incentives for the receiver and network node to cooperate honestly.

Accuracy

Classic ECN feeds back one congestion notification per RTT, as this is supposed to be used for TCP congestion control which reduces the sending rate at most once per RTT. The accurate ECN feedback scheme has to ensure that if a congestion events occurs at least one congestion notification is echoed and received per RRT as classic ECN would do. Of course, the goal of this extension is to reconstruct the number of CE marking more accurately. However, a sender should not assume to get the exact number of congestion marking in a high congestion situation.

Complexity
Of course, the more accurate ECN feedback can also be used, even if only one ECN feedback signal per RTT is need. To enable this proposal for a more accurate ECN feedback as the standard ECN feedback mechanism, the implementation should be as simple as possible and a minimum of addition state information should be needed.

3.1.2. One bit feedback flag

Remark: In one Acknowledgment all acknowledged bytes are regarded as congested

This option is using a one bit flag, namely the ECE bit, to signal more accurate ECN feedback. Other than classic ECN feedback, a accurate ECN feedback receiver MUST set the ECE bit only in one ACK packets for each one CE received. An more accurate ECN feedback receiver MUST NOT wait for a CWR bit from the sender to reset the ECE bit.

As the CWR would now be unused, the CWR MUST be set in the subsequent ACK after the ECE was set.

CWR(t) = ECE(t-1)

This provides some redundancy in case of ACK loss. If the sender know the ACK'ing scheme of the receiver (e.g. delayed ACKs will send minimum one ACK for every two data packets), the sender can detect ACK loss. If two subsequent ACK or more got lost, the sender SHOULD assume congestion marks for the respective number of ack'ed bytes.

Moreover, when a congestion situation occurs or stops, the receiver MUST immediately acknowledge the data packet and MUST NOT delay the acknowledgment until a further data packet is arrived. A congestion situation occurs when the previous data packet was CE=0 but the current one is CE=1. And a congestion situation stops when the previous data packet was CE=1 and the current one is CE=0.

The following figure shows a simple state machine to describe the receiver behavior.

```
                             Send immediate
                             ACK with ECE=0
                       .---.    .------------.     .---.
         Send 1 ACK  /     v    v            |    |     \
          for every |     .------.          .------.     | Send 1 ACK
          m packets |     | CE=0 |          | CE=1 |     | for every
         with ECE=0 |     '------'          '------'     | m packets
                     \     |    |            ^    ^     /  with ECE=1
                       '---'    '------------'     '---'
                              Send immediate
                              ACK with ECE=1
```

Figure 2: Two state ACK generation state machine

Thus whenever an ACK with the ECE flag set arrives, all acknowledged byte were congestion marked. This scheme provides a byte-wise ECN feedback. The number of CE-marked packet can be estimated by dividing the amount of ack'ed bytes by the Maximum Segment Size (MSS).

When one ACK was lost and the ECN feedback is received based on the CWR set, the sender conservatively SHOULD assume all newly acked bytes as congestion marked.

3.1.2.1. Discussion

ACK loss

In low congestion situations (less than one CE mark per RTT on average), the loss of two subsequent ACKs would result in complete loss of the congestion information. The opposite would be true during high congestion, where the sender can incorrectly assume that all segments were received with the CE codepoint.

One solution would be to carry the same information in a defined number of subsequent ACK packets. This would reduce the number of feedback signals that can be transmitted in one RTT but improve the integrity. More sophisticated solutions based on ACK loss detection might be possible as well.

With DCTCP [Ali10] it was proposed to acknowledge a data packet directly without delay when a congestion situation occurs, as already described above. This scheme allows a more accurate feedback signal in a high congestion/marking situation. However, using delayed ACKs is important for a variety of reasons, including reducing the load on the data sender.

As this heuristic is triggering immediate ACKs whenever the received

CE bit toggles, arbitrarily large ACK ratios are supported. However, the effective ACK ratio is depending on the congestion state of the network. Thus it may collapse to 1 (one ACK for each data segment). More sophisticated solutions based on ACK loss detection might be possible as well, when every other segment is received with CE set.

ECN Nonce

As the ECN Nonce bit is not used otherwise, ECN Nonce [RFC3540] can be used complementary. Network paths not supporting ECN, misbehaving, or malicious receivers withholding ECN information can therefore be detected.

3.1.3. Three bit field with counter feedback

The receiver maintains an unsigned integer counter which we call ECC (echo congestion counter). This counter maintains a count of how many times a CE marked packet has arrived during the half-connection. Once a TCP connection is established, the three TCP option flags (ECE, CWR and NS) are used as a 3-bit field for the receiver to permanently signal the sender the current value of ECC, modulo 8, whenever it sends a TCP ACK. We will call these three bits the echo congestion increment (ECI) field.

This overloaded use of these 3 option flags as one 3-bit ECI field is shown in Figure 3. The actual definition of the TCP header, including the addition of support for the ECN Nonce, is shown for comparison in Figure 1. This specification does not redefine the names of these three TCP option flags, it merely overloads them with another definition once a flow with accurate ECN feedback is established.

```
  0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               |           |           | U | A | P | R | S | F |
| Header Length | Reserved  |    ECI    | R | C | S | S | Y | I |
|               |           |           | G | K | H | T | N | N |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

Figure 3: Definition of the ECI field within bytes 13 and 14 of the TCP Header (when SYN=0).

Also note that, whenever the SYN flag of a TCP segment is set (including when the ACK flag is also set), the NS, CWR and ECE flags (i.e. the ECI field of the SYNACK) MUST NOT be interpreted as the 3-bit ECI value, which is only set as a copy of the local ECC value in non-SYN packets.

This scheme was first proposed in [I-D.briscoe-tsvwg-re-ecn-tcp] for the use with re-ECN.

3.1.3.1. Discussion

ACK loss

As pure ACKs are not protected by TCP reliable delivery, we repeat the same ECI value in every ACK until it changes. Even if many ACKs in a row are lost, as soon as one gets through, the ECI field it repeats from previous ACKs that didn't get through will update the sender on how many CE marks arrived since the last ACK got through.

The sender will only lose a record of the arrival of a CE mark if all the ACKS are lost (and all of them were pure ACKs) for a stream of data long enough to contain 8 or more CE marks. So, if the marking fraction was p, at least 8/p pure ACKs would have to be lost. For example, if p was 5%, a sequence of 160 pure ACKs (without delayed ACKs) would all have to be lost. When ACK are delay this number has to be reduced by 1/m. This would still require a sequence of 80 pure lost ACKs with the usual delay rate of m=2.

Additionally, to protect against such extremely unlikely events, if a re- ECN sender detects a sequence of pure ACKs has been lost it can assume the ECI field wrapped as many times as possible within the sequence. E.g., if a re-ECN sender receives an ACK with an acknowledgement number that acknowledges L (>m) segments since the previous ACK but with a sequence number unchanged from the previously received ACK, it can conservatively assume that the ECI field incremented by D' = L - ((L-D) mod 8), where D is the apparent increase in the ECI field. For example if the ACK arriving after 9 pure ACK losses apparently increased ECI by 2, the assumed increment of ECI would still be 2. But if ECI apparently increased by 2 after 11 pure ACK losses, ECI should be assumed to have increased by 10.

ECN Nonce

ECN Nonce cannot be used in parallel to this scheme. But mechanism that make use of this new scheme might provide stronger incentives to declare congestion honestly when needed. E.g. with ConEx each congestion notification suppressed by the receiver should lead the ConEx audit function to discard an equivalent number of bytes such that the receiver does not gain from suppressing feedback. This mechanism would even provide a stronger integrity mechanism than ECN-Nonce does. Without an external framework to discourage the withholding of ECN information, this scheme is vulnerable to the problems described in [RFC3540].

3.1.4. Codepoints with dual counter feedback

In-line with the definition of the previous section in Figure 3, the ECE, CWR and NS bits are used as one field but instead they are encoding 8 codepoints. These 8 codepoints, as shown below, encode either a "congestion indication" (CI) counter or an ECT(1) counter (E1). These counters maintain the number of CE marks or the number of ECT(1) signals observed at the receiver respectively.

ECI	NS	CWR	ECE	CI (base5)	E1 (base3)
0	0	0	0	0	-
1	0	0	1	1	-
2	0	1	0	2	-
3	0	1	1	3	-
4	1	0	0	4	-
5	1	0	1	-	0
6	1	1	0	-	1
7	1	1	1	-	2

Table 2: Codepoint assignment for accurate ECN feedback

By default an accurate ECN receiver MUST echo the CI counter (modulo 5) with the respective codepoints. Whenever an CE occurs and thus the value of the CI has changed, the receiver MUST echo the CI in the next ACK. Moreover, the receiver MUST repeat the codepoint, that provides the CI counter, directly on the subsequent ACK. Thus every value of CI will be transmitted at least twice.

If an ECT(1) mark is receipt and thus E1 increases, the receiver has to convey that updated information to the sender as soon as possible. Thus on the reception of a ECT(1) marked packet, the receiver MUST signal the current value of the E1 counter (modulo 3) in the next ACK, unless a CE mark was receipt which is not echoed yet twice. The receiver MUST also repeat very E1 value. But this repetition does not need to be in the subsequent ACK as the E1 value will only be transmitted when no changes in the CI have occured. Each E1 value will be send excatly twice. The repetition of every signal will provide further resilience against lost ACKs.

As only a limited number of E1 codepoints exist and the receiver might not acknowledge every single data packet immediately (delayed ACKs), a sender SHOULD NOT mark more than 1/m of the packets with ECT(1), where m is the ACK ratio (e.g. 50% when every second data packet triggers an ACK). This constraint will avoid a permanent feedback of E1 only.

This requirement may conflict with delayed ACK ratios larger than two, using the available number of codepoints. A receiver MUST change the ACK'ing rate such that a sufficient rate of feedback signals can be sent. Details on how the change in the ACK'ing rate should be implemented are given in the next subsection.

3.1.4.1. Implementation

The basic idea is for the receiver to count how many packets carry a congestion notification. This could, in principle, be achieved by increasing a "congestion indication" counter (CI.c) for every incoming CE marked segment. Since the space for communicating the information back to the sender in ACKs is limited, instead of directly increasing this counter, a "gauge" (CI.g) is increased instead.

When sending an ACK, the content of this gauge (capped by the maximum number that can be encoded in the ACK, e.g. 4 for CI, and 2 for E1) is copied to the actual counter, and CI.g is reduced by the value that was copied over and transmitted, unless CI.g was zero before. To avoid losing information, it is ensured that an ACK is sent at least after 5 incoming congestion marks (i.e. when CI.g exceeds 5).

For resilience against lost ACKs, an indicator flag (CI.i) ensures that, whether another congestion indication arrives or not, a second ACK transmits the previous counter value again.

The same counter / gauge method is used to count and feed back (using a different mapping) the number of incoming packets marked ECT(1) (called E1 in the algorithm). As fewer codepoints are available for conveying the E1 counter value, an immediate ACK MUST be triggered whenever the gauge E1.g exceeds a threshold of 3. The sender receives the receiver's counter values and compares them with the locally maintained counter. Any increase of these counters is added to the sender's internal counters, yielding a precise number of CE-marked and ECT(1) marked packets. Architecturally the counters never decrease during a TCP session. However, any overflow must be modulo 5 for CI, and modulo 3 for E1.

The following table provides an example showing an half-connection with an TCP sender A and receiver B. The sender maintains a counter CI.r to reconstruct the number of CE mark receipt at receiver-side.

	Data	TCP A	IP	TCP B	Data
		SEQ ACK CTL		SEQ ACK CTL	
--		-------------	----------	-------------	
1		0100 SYN	---->		
		CWR,ECE,NS			
2			<----	0300 0101 SYN	
				ACK,CWR	
3		0101 0301 ACK	ECT0 -CE->		
				CI.c=0 CI.g=1	
4	100	0101 0301 ACK	ECT0 ---->		
				CI.c=1 CI.g=0	
5			<----	0301 0201 ACK	
				ECI=CI.1	
		CI.r=1			
6	100	0201 0301 ACK	ECT0 -CE->		
				CI.c=1 CI.g=1	
7	100	0301 0301 ACK	ECT0 -CE->		
				CI.c=1 CI.g=2	
8			XX--	0301 0401 ACK	
				ECI=CI.1	
		CI.r=1			
9	100	0401 0301 ACK	ECT0 -CE->		
				CI.c=1 CI.g=3	
10	100	0501 0301 ACK	ECT0 -CE->		
				CI.c=5 CI.g=0	
11			<----	0301 0601 ACK	
				ECI=CI.0	
		CI.r=5			
12	100	0601 0301 ACK	ECT0 -CE->		
				CI.c=5 CI.g=1	
13	100	0701 0301 ACK	ECT0 -CE->		
				CI.c=5 CI.g=2	
14			<----	0301 0801 ACK	
				ECI=CI.0	
		CI.r=5			

Table 3: Codepoint signal example

3.1.4.2. Discussion

ACK loss

As this scheme sends each codepoint (of the two subsets) at least two times, at least one, and up to two consecutive ACKs can be lost. Further refinements, such as interleaving ACKs when sending codepoints belonging to the two subsets (e.g. CI, E1), can allow the loss of any two consecutive ACKs, without the sender losing congestion information, at the cost of also reducing the ACK ratio.

At low congestion rates, the sending of the current value of the CI counter by default allows higher numbers of consecutive ACKs to be lost, without impacting the accuracy of the ECN signal.

ECN Nonce

By comparing the number of incoming ECT(1) notifications with the actual number of packets that were transmitted with an ECT(1) mark as well as the sum of the sender's two internal counters, the sender can probabilistic detect a receiver that would send false marks or supress accurate ECN feedback, or a path that doesn't properly support ECN.

This approach maintains a balanced selection of properties found in ECN Nonce, Section 3.1.3, and Section 3.1.2. A delayed ACK ratio of two can be sustained indefinitely even during heavy congestion, but not during excessive ECT(1) marking, which is under the control of the sender. An higher ACK ratios can be sustained even when congestion is low but its need for the E1 feedback.

3.1.5. Short Summary of the Discussions

With the exception of the signaling scheme described in Section 3.1.2, all signaling may fail to work, if middleboxes intervene and check on the semantic of [RFC3168] signals.

The scheme described in Section 3.1.4 is the most complex to implement especially on a receiver, with much additional state to be kept there, compared to the other signaling schemes. With the advances in compute power, many more cycles are available to process TCP than ever before.

Table 4 gives an overview of the relative implications of the different proposed signaling schemes. Further discussion should be included here in the next version of this document.

Section	Resi- liency	Timely	Integrity	Accuracy	Complexity
1-bit-flag	-	+	+	-	+
3-bit-field	++	++	--	++	-
Codepoints	+	+	+	++	--

Table 4: Overview of accurate feedback schemes

Whereas the first scheme is the simplest one (and also provides byte-wise feedback which might be preferable), it has a drawback with respect to reliability. The second one is the most reliable but does not provide an integrity mechanism.

3.2. TCP Sender

This section will specify the sender-side action describing how to exclude the accurate number of congestion markings from the given receiver feedback signal.

When the accurate ECN feedback scheme is supported by the receiver, the receiver will maintain an echo congestion counter (ECC). The ECC will hold the number of CE marks received. A sender that is understanding the accurate ECN feedback will be able to reconstruct this ECC value on the sender side by maintaining a counter ECC.r.

On the arrival of every ACK, the sender calculates the difference D between the local ECC.r counter, and the signaled value of the receiver side ECC counter. The value of ECC.r is increased by D, and D is assumed to be the number of CE marked packets that arrived at the receiver since it sent the previously received ACK.

3.3. TCP Receiver

This section will describe the receiver-side action to signal the accurate ECN feedback back to the sender. In any case the receiver will need to maintain a counter of how many CE marking has been seen during a connection. Depending on the chosen coding scheme there will be different action to set the corresponding bits in the TCP header. For all case it might be helpful if the receiver is able to switch form a delayed ACK behavior to send ACKs immediately after the data packet reception in a hight congestion situation.

3.4. Advanced Compatibility Mode

This section describes a possiblity to achieve more accurate feedback even when the receiver is not capable of the new accurate ECN feedback scheme with the drawback of less reliability.

During initial deployment, a large number of receivers will only support [RFC3168] classic ECN feedback. Such a receiver will set the ECE bit whenever it receives a segment with the CE codepoint set, and clear the ECE bit only when it receives a segment with the CWR bit set. As the CE codepoint has priority over the CWR bit (Note: the wording in this regard is ambiguous in [RFC3168], but the reference implementation of ECN in ns2 is clear), a [RFC3168] compliant receiver will not clear the ECE bit on the reception of a segment, where both CE and CWR are set simultaneously. This property allows the use of a compatibility mode, to extract more accurate feedback from legacy [RFC3168] receivers by setting the CWR permanently.

Assuming an delayed ACK ratio of one, a sender can permanently set the CWR bit in the TCP header, to receive a more accurate feedback of the CE codepoints as seen at the receiver. This feedback signal is however very brittle and any ACK loss may cause congestion information to become lost. Delayed ACKs and ACK loss can both not be accounted for in a reliable way, however. Therefore, a sender would need to use heuristics to determine the current delay ACK ratio m used by the receiver (e.g. most receivers will use m=2), and also the recent ACK loss ratio (l). Acknowledge Congestion Control (AckCC) as defined in [RFC5690] can not be used, as deployment of this feature is only experimental.

Using a phase locked loop algorithm, the CWR bit can then be set only on those data segments, that will trigger a (delayed) ACK. Thereby, no congestion information is lost, as long as the ACK carrying the ECE bit is seen by the sender.

Whenever the sender sees an ACK with ECE set, this indicates that at least one, and at most m / (m - l) data segments with the CE codepoint set where seen by the receiver. The sender SHOULD react, as if m CE indications where reflected back to the sender by the receiver, unless additional heuristics (e.g. dead time correction) can determine a more accurate value of the "true" number of received CE marks.

4. Acknowledgements

We want to thank Michael Welzl and Bob Briscoe for their input and discussion.

5. IANA Considerations

This memo includes no request to IANA.

6. Security Considerations

For coding schemes that increase robustness for the ECN feedback, similar considerations as in RFC3540 apply for the selection of when to sent a ECT(1) codepoint.

7. References

7.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC3540] Spring, N., Wetherall, D., and D. Ely, "Robust Explicit Congestion Notification (ECN) Signaling with Nonces", RFC 3540, June 2003.

7.2. Informative References

[Ali10] Alizadeh, M., Greenberg, A., Maltz, D., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., and M. Sridharan, "DCTCP: Efficient Packet Transport for the Commoditized Data Center", Jan 2010.

[I-D.briscoe-tsvwg-re-ecn-tcp] Briscoe, B., Jacquet, A., Moncaster, T., and A. Smith, "Re-ECN: Adding Accountability for Causing Congestion to TCP/IP", draft-briscoe-tsvwg-re-ecn-tcp-09 (work in progress), October 2010.

[RFC5562] Kuzmanovic, A., Mondal, A., Floyd, S., and K. Ramakrishnan, "Adding Explicit Congestion Notification (ECN) Capability to TCP's SYN/ACK Packets", RFC 5562, June 2009.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.

[RFC5690] Floyd, S., Arcia, A., Ros, D., and J. Iyengar, "Adding Acknowledgement Congestion Control to TCP", RFC 5690, February 2010.

Appendix A. Pseudo Code for the Codepoint Coding

Receiver:

Input signals: CE , ECT(1)
TCP Fields: ECI (3-bit field from CWR and ECE). CI.cm and E1.cm map into these 8 codepoints (ie. 5 and 3 codepoints)

These counters get tracked by the following variables:

```
CI.c (congestion indication - counter, modulo a multiple of the
available codepoints to represent CI.c in the ECI field.
Range[0..n*CI.cp-1])
CI.g (congestion indication - gauge, [0.."inf"])
CI.i (congestion indication - iteration, [0,1])
These are to track CE indications.

E1.c, E1.g and E1.r (doing the same, but for ECT(1) signals).

Constants:
CI.cp (number of codepoints available to signal)
CI.cm[] (codepoint mapping for CI)
E1.cp (number of codepoints available for E1 signal)
E1.cm[0..(E1.cp-1)] (codepoint mappings for E1)

At session initialization, all these counters are set to 0;

When a Segement (Data, ACK) is received,
perform the following steps:

If a CE codepoint is received,
Increase CI.g by 1
If a ECT(1) codepoint is received,
Increase E1.g by 1
If (CI.g > 5)         # When ACK rate is not sufficient to keep
or (E1.g > 3)         # gauge close to zero, increase ACK rate
# works independent of delACK number (ie AckCC)
Cancel pending delayed ACK (ACK this segment immediately)
# this increases the ACK rate to a maximum of 1.5 data segments
# per ACK, with delACK=2,
# and CE mark rate exceeds 75% for a number
# of at least 18 segments.
# 5 codepoints would allow delack=2 indefinitely btw

When preparing an ACK to be sent:

If (CI.g > 0) or
((E1.i != 0) and (CI.i != 0))  # E1.g = 0 is to skip this
                               # if only the 2nd CI.c ACK
# has to be sent - effectively alternating CI.c and E1.c on ACKs
# should give slightly better resiliency against ack losses
If CI.i == 0                   # updates to CI.c allowed
and CI.g > 0                   # update is meaningful
CI.i = 1                       # may be larger
                               #if more resiliency is reqd
CI.c += min(CI.cp-1,CI.g)      # CI.cp-1 is 3 for 4 codepoints,
```

```
                                  # 4 for 5 etc
   CI.c = CI.c modulo CI.cp*CI.cp # using modulo the square of
                                  # available codepoints,
                                  # for convinience (debugging)
   CI.g -= min(CI.cp-1,CI.g)      #
   Else
   CI.i--                         # just in case CI.f was set to
                                  # more than 1 for resiliency
   Send next ACK with ECI = CI.cm[CI.c modulo CI.cp]
   Else
   If (E1.g > 0) or (E1.i != 0)

   If (E1.i == 0) and (E1.g > 0)
   E1.i = 1
   E1.c += min(E1.cp-1,E1.g)
   E1.c = E1.c modulo E1.cp*E1.cp
   E1.g -= min(E1.cp-1,E1.g)
   Else
   E1.i--
   Send next ACK with ECI = E1.cm[E1.c modulo E1.cp]
   Else
   Send next ACK with ECI = CI.cm[CI.c modulo CI.cp] # default action
```

Sender:

Counters:

```
   CI.r - current value of CEs seen by receiver
   E1.s - sum of all sent ECT(1) marked packets (up to snd.nxt)
   E1.s(t) - value of E1.s at time (in sequence space) t
   E1.r - value signaled by receiver about received ECT(1) segments
   E1.r(t) - value of E1.r at time (in sequence space) t
   CI.r(t) - ditto
```

```
# Note: With a codepoint-implementation,
# a reverse table ECI[n] -> CI.r / E1.r is needed.
# This example is simplified with 4/4 codepoints
# instead of 5/3

If ACK with NS=0
CI.r +=  (ECI + 4 - (CI.r mod CI.cp)) mod CI.cp
# The wire protocol transports the absolute value
# of the receiver-side counter.
# Thus the (positive only) delta needs to be calculated,
# and added to the sender-side counter.
If ACK with NS=1
E1.r += (ECI + 4 - (E1.r mod E1.cp)) mod E1.c

# Before CI.r or E1.r reach a (binary) rollover,
# they need to roll over some multiple of CI.cp
# and E1.cp respectively.

CI.r = CI.r modulo CI.cp * n_CI
E1.r = E1.r modulo E1.cp * n_E1

# (an implementation may choose to use a single constant,
# ie 3^4*5^4 for 16-bit integers,
# or 3^8*5^8 for 32-bit integers)

# The following test can (probabilistically) reveal,
# if the receiver or path is not properly
# handling ECN (CE, E1) marks

If not E1.r(t) &lt;= E1.s(t) &lt;= E1.r(t) + CI.r(t)
# -> receiver lies (or too many ACKs got lost,
# which can be checked too by the sender).
```

Authors' Addresses

Mirja Kuehlewind (editor)
University of Stuttgart
Pfaffenwaldring 47
Stuttgart 70569
Germany

Email: mirja.kuehlewind@ikr.uni-stuttgart.de

Richard Scheffenegger
NetApp, Inc.
Am Euro Platz 2
Vienna,   1120
Austria

Phone: +43 1 3676811 3146
Email: rs@netapp.com

Congestion Exposure (ConEx) M. Kuehlewind, Ed.
Internet-Draft University of Stuttgart
Intended status: Experimental R. Scheffenegger
Expires: May 3, 2012 NetApp, Inc.
October 31, 2011

TCP modifications for Congestion Exposure
draft-kuehlewind-conex-tcp-modifications-01

Abstract

Congestion Exposure (ConEx) is a mechanism by which senders inform the network about the congestion encountered by previous packets on the same flow. This document describes the necessary modifications to use ConEx with the Transmission Control Protocol (TCP).

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction 3
1.1. Requirements Language 3
2. Sender-side Modifications 3
3. Accounting congestion 4
3.1. ECN 5
3.1.1. Accurate ECN feedback 5
3.1.2. Classic ECN support 5
3.2. Loss Detection with/without SACK 7
4. Setting the ConEx IPv6 Bits 7
4.1. Setting the E and the L Bit 8
4.2. Credit Bits 8
5. Timeliness of the ConEx Signals 9
6. Acknowledgements 10
7. IANA Considerations 10
8. Security Considerations 10
9. References 10
9.1. Normative References 10
9.2. Informative References 11
Authors' Addresses 11

1. Introduction

Congestion Exposure (ConEx) is a mechanism by which senders inform the network about the congestion encountered by previous packets on the same flow. This document describes the necessary modifications to use ConEx with the Transmission Control Protocol (TCP). The ConEx signal is based on loss or ECN marks [RFC3168] as a congestion indication.

With standard TCP without Selective Acknowledgments (SACK) [RFC2018] the actual number of losses is hard to detect, thus we recommend to enable SACK when using ConEx. However, we discuss both cases, with and without SACK support, later on.

Explicit Congestion Notification (ECN) is defined in such a way that only a single congestion signal is guaranteed to be delivered per Round-trip Time (RTT). For ConEx a more accurate feedback signal would be beneficial. Such an extension to ECN is defined in a seperate document [draft-kuehlewind-conex-accurate-ecn], as it can also be useful for other mechanisms, as e.g. [DCTCP] or whenever the congestion control reaction should be proportional to the expirienced congestion.

ConEx is currently/will be defined as an destination option for IPv6. The use of four bits have been defined, namely the X (ConEx-capable), the L (loss experienced), the E (ECN experienced) and C (credit) bit.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Sender-side Modifications

A ConEx sender MUST negotitate for both SACK and the more accurate ECN feedback in the TCP handshake if these TCP extension are available at the sender. Depending on the capability of the receiver, the following operation modes exist:

- Full-ConEx (SACK and accurate ECN feedback)

- accECN-ConEx (no SACK but accurate ECN feedback)

- ECN-ConEx (no SACK and no accurate ECN feedback but 'classic' ECN)

- o SACK-ECN-ConEx (SACK and 'classic' instead of accurate ECN)
- o SACK-ConEx (SACK but no ECN at all)
- o Basic-ConEx (neither SACK nor ECN)

A ConEx sender MUST expose congestion to the network according to the congestion information received by ECN or based on loss provided by the TCP feedback loop. A TCP sender MUST account congestion byte-wise (and not packet-wise) and MUST mark the respective number of payload bytes in subsequent packets (after the congestion notification) with the respective ConEx bit in the IP header. The congestion accounting based on different operation modes is described in the next section and the handling of the IPv6 bits itself in the subsequent section afterwards.

3. Accounting congestion

A TCP sender MUST account congestion byte-wise (and not packet-wise) based the congestion information received by ECN or loss detection provided by TCP. For this purpose a TCP sender will maintain two different counters for number outstanding bytes that need to be ConEx marked either with the E bit or the L Bit.

The outstanding bytes accounted based on ECN feedback information are maintained in the congestion exposure gauge (CEG). The accounting of these bytes from the ECN feedback is explained in more detail next.

The outstanding bytes for congestion indications based on loss are maintained in the loss exposure gauge (LEG) and the accounting is explained in subsequent to the CEG accounting.

The subtraction of bytes which have been ConEx marked from both counters is explained in the next section.

Usually all byte of an IP packet must be accounted. If we assume equal sized packets or at least equally distributed packet sizes the sender MAY only account the TCP payload bytes, as the ConEx marked packets as well as the original packets causing the congestion will both contain about the same number of headers. Otherwise the sender MUST take the headers into account. A sender which sends different sized packets with unequally distributed packet sizes should know about reason to do so and thus may be able to reconstruct the exact number of headers based on this information. Otherwise if no additional information is available the worse case number of headers SHOULD be estimated in a conservative way based on a minimum packet size (of all packets sent in the last RTT).

3.1. ECN

A receiver can support the accurate ECN feedback scheme, the 'classic' ECN or neither. In the case ECN is not supported at all, the transport is not ECN-capable and no ECN marks will occur, thus the E bit will never be set. In the other cases a ConEx sender MUST maintain a gauge for the number of outstanding bytes that has to be ConEx marked with the E bit, the congestion exposure gauge (CEG).

The CEG is increased when ECN information is received from an ECN-capable receiver supporting the 'classic' ECN scheme or the accurate ECN feedback scheme. When the ConEx sender receives an ACK indicating one or more segments were received with a CE mark, CEG is increased by the appropriate number of bytes. The two cases, depending on the receiver capability, are discussed in the following sections.

3.1.1. Accurate ECN feedback

With an more accurate ECN feedback scheme either the number of marked packets/received CE marks is know or the number of marked bytes directly. In the later case the CEG can directly be increased by the number of marked bytes. Otherwise when the accurate ECN feedback scheme is supported by the receiver, the receiver will maintain an echo congestion counter (ECC). The ECC will hold the number of CE marks received. A sender that is understanding the accurate ECN feedback will be able to reconstruct this ECC value on the sender side by maintaining a counter ECC.r.

On the arrival of every ACK, the sender calculates the difference D between the local ECC.r counter, and the signaled value of the receiver side ECC counter. The value of ECC.r is increased by D, and D is assumed to be the number of CE marked packets that arrived at the receiver since it sent the previously received ACK.

Whenever the counter ECC.r is increased, the gauge CEG has to be increased by the amount of bytes sent which were marked:

```
CEG += min( SMSS*D, acked_bytes )
```

3.1.2. Classic ECN support

A ConEx sender that communicates with a classic ECN receiver (conforming to [RFC3168] or [RFC5562]) MAY run in one of these modes:

- o Full compliance mode:

  The ConEx sender fully conforms to all the semantics of the ECN signaling as defined by [RFC5562]. In this mode, only a single congestion indication can be signaled by the receiver per RTT. Whenever the ECE flag toggles from "0" to "1", the gauge CEG is increased by the SMSS:

CEG += SMSS

Note that under severe congestion, a session adhering to these semantics may not provide enough ConEx marks. This may cause appropriate sanctions by an audit device in a ConEx enabled network.

- Simple compatibility mode:

  The sender will set the CWR permanently to force the receiver to signal only one ECE per CE mark. Unfortunately, in a high congestion situation where all packets are CE marled over a certain period of time, the use of delayed ACKs, as it is usually done today, will prevent a feedback of every CE mark. With an ACK rate of m, about m-1/m CE indications will not be signaled back by the receiver (e.g. 50% with M=2). Thus, in this mode the ConEx sender MUST increase CEG by a count of M*SMSS for each received ECE signal:

  CEG += M*SMSS

  In case of a congestion event with low congestion (that means when only a very smaller number of packets get marked), the sender might miss the whole congestion event. In average the sender will sent sufficient ConEx marks due to the scheme proposed above but these ConEx marks might be timely shifted. Regarding congestion control it is not a general problem to miss a congestion event as by chance a marking scheme in the network node might also miss a certain flow. Even if then no other flow is reacting, the congestion level will increase and it will get more likely that the congestion feedback is delivered. But to provide a fair share over time, a TCP sender could react more strong when receiving a ECN feedback signal. This of course depends on the congestion control used. A TCP sender using this scheme MUST take the impact on congestion control into account.

- Advanced compatibility mode:

  More sophisticated heuristics, such as a phase locked loop, to set CWR only on those data segments, that will actually trigger an (delayed) ACK, could extract congestion notifications more timely. A ConEx sender MAY choose to implement such a heuristic. In addition, further heuristics SHOULD be implemented, to determine the value of each ECE notification. E.g. for each consecutive ACK received with the ECE flag set, CEG should be increased by min(M*SSMS, acked_bytes). Else if the predecessor ACK was received with the ECE flag cleared, CEG need only be increase by one SMSS:

```
if previous_marked: CEG += min( M*SSMS, acked_bytes)
else: CEG += SMSS
```

This heuristic is conservative during more serious congestion, and more relaxed at low congestion levels.

3.2. Loss Detection with/without SACK

For all the data segments that are determined by a ConEx sender as lost, an identical number of IP bytes MUST be be sent with the ConEx L bit set. Loss detection typically happens by use of duplicate ACKs, or the firing of the retransmission timer. A ConEx sender MUST maintain a loss exposure gauge (LEG), indicating the number of outstanding bytes that must be sent with the ConEx L bit. When a data segment is retransmitted, LEG will be increased by the size of the TCP payload packet containing the retransmission, assuming equal sized segments such that the retransmitted packet will have the same number of header as the original ones. When sending subsequent segments (including TCP control segments), the ConEx L bit is set as long as LEG is positive, and LEG is decreased by the size of the sent TCP payload with the ConEx L bit set.

Any retransmission may be spurious. To accommodate that, a ConEx sender SHOULD make use of heuristics to detect such spurious retransmissions (e.g. F-RTO [RFC5682], DSACK [RFC3708], and Eifel [RFC3522], [RFC4015]). When such a heuristic has determined, that a certain number of packets were retransmitted erroneously, the ConEx sender should subtract the payload size of these TCP packets from LEG.

Note that the above heuristics delays the ConEx signal by one segment, and also decouples them from the retransmissions themselves, as some control packets (e.g. pure ACKs, window probes, or window updates) may be sent in between data segment retransmissions. A simpler approach would be to set the ConEx signal for each retransmitted data segment. However, it is important to remember, that a ConEx signal and TCP segments do not natively belong together.

4. Setting the ConEx IPv6 Bits

ConEx is currently/will be defined as an destination option for IPv6. The use of four bits have been defined, namely the X (ConEx-capable), the L (loss experienced), the E (ECN experienced) and C (credit) bit.

By setting the X bit a packet is marked as ConEx-capable. All packets carrying payload MUST be marked with the X bit set including retransmissions. About control packets as pure ACKs which are not carrying any payload no congestion feedback information are available thus these packet should not be take into account when determining ConEx information. These packet MUST carry a ConEx Destination Option with the X bit unset.

4.1. Setting the E and the L Bit

As long as the CEG/LEG is positive, ConEx-capable packets MUST be marked with E or respective L and the CEG/LEG is decreased by the TCP payload bytes carried in this packet. If the CEG/LEG is negative, the CEG/LEG is drained by one byte with every packet sent out, as ConEX information are only meaningful for a certain time:

```
if CEG > 0: CEG -= TCPpayload.length else: CEG--
if LEG > 0: LEG -= TCPpayload.length else: LEG--
```

4.2. Credit Bits

The ConEx abstract mechanism requires that the transport SHOULD signal sufficient credit in advance to cover any reasonably expected congestion during its feedback delay. To be very conservative the number of credits would need to equal the number of packets in flight, as every packet could get lost or congestion marked. With a more moderate view, only an increase in the sending rate should cause congestion.

For TCP sender using the [RFC5681] congestion control algorithm, we recommend to only send credit in Slow Start, as in Congestion Avoidance an increase of one segment per RTT should only cause a minor amount of congestion marks (usually at max one). If a more aggressive congestion control is used, a sufficient amount of credits need to be set.

In TCP Slow Start the sending rate will increase exponentially and that means double every RTT. Thus the number of credits should equal half the number of packets in flight in every RTT. Under the assumption that all marks will not get invalid for the whole Slow Start phase, marks of a previous RTT have to be summed up. Thus the marking of every fourth packet will allow sufficient credits in Slow Start.

```
                RTT1    |------XC------>|
                        |------X------->|
                        |------X------->|   credit=1  in_flight=3
                        |               |
                RTT2    |------X------->|
                        |------XC------>|
                        |------X------->|
                        |------X------->|
                        |------X------->|
                        |------XC------>|   credit=3  in_flight=6
                        |               |
                RTT3    |------X------->|
                        |------X------->|
                        |------X------->|
                        |------XC------>|
                        |------X------->|
                        |------X------->|
                        |------X------->|
                        |------XC------>|
                        |------X------->|
                        |------X------->|
                        |------X------->|
                        |------XC------>|   credit=6  in_flight=12
                        |       .       |
                        |       :       |
```

Figure 1: Credits in Slow Start (with an initial window of 3)

If a ConEx sender detects an increasing number of losses even though the sender reduced the sending rate, the sender SHOULD assume that those losses are incorporated by an audit device and thus should send further credits. Up to now its not clear if the credits say valid as long as the connection is established or if an expiration of the credits need to be assumed by the sender.

5. Timeliness of the ConEx Signals

ConEx signals will anyway be evaluated with a slight time delay of about one RTT by a network node. Therefore, it would not be absolutely necessary to immediately signal ConEx bits when they become known (e.g. L and E bits), but a sender SHOULD sent the ConEx signaling with the next available packet. If cases are available where it is preferable to slight delay the ConEx signal, the sender MUST NOT delay the ConEx signal more than one RTT.

Multiple ConEx bits may become available for signaling at the same time, for example when an ACK is received by the sender, that indicates that at least one segment has been lost, and that one or more ECN marks were received at the same time. This may happen during excessive congestion, where buffer queues overflow and some packets are marked, while others have to be dropped nevertheless. Another possibility when this may happen are lost ACKs, so that a subsequent ACK carries summary information not previously available to the sender.

It is important to remember, that ConEx bits and TCP retransmissions do not interact with each other. However, a retransmission should be accompanied by one ConEx L bit in close proximity nevertheless. This does not mean, that TCP retransmissions may never contain ConEx marks. In a typical scenario using SACK, the first retransmission would not carry a ConEx L bit, while subsequent retransmissions in the same recovery episode, would be marked with the ConEx L bit. Spreading the ConEx bits over a small number of segments increases the likelihood that most devices along the path will see some ConEx marks even during heavy congestion.

6. Acknowledgements

7. IANA Considerations

8. Security Considerations

9. References

9.1. Normative References

[RFC2018] Mathis, M., Mahdavi, J., Floyd, S., and A. Romanow, "TCP Selective Acknowledgment Options", RFC 2018, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC5562] Kuzmanovic, A., Mondal, A., Floyd, S., and K. Ramakrishnan, "Adding Explicit Congestion Notification (ECN) Capability to TCP's SYN/ACK Packets", RFC 5562, June 2009.

9.2. Informative References

[DCTCP] Alizadeh, M., Greenberg, A., Maltz, D., Padhye, J., Patel, P., Prabhakar, B., Sengupta, S., and M. Sridharan, "DCTCP: Efficient Packet Transport for the Commoditized Data Center", Jan 2010.

[I-D.briscoe-tsvwg-re-ecn-tcp] Briscoe, B., Jacquet, A., Moncaster, T., and A. Smith, "Re-ECN: Adding Accountability for Causing Congestion to TCP/IP", draft-briscoe-tsvwg-re-ecn-tcp-09 (work in progress), October 2010.

[RFC3522] Ludwig, R. and M. Meyer, "The Eifel Detection Algorithm for TCP", RFC 3522, April 2003.

[RFC3708] Blanton, E. and M. Allman, "Using TCP Duplicate Selective Acknowledgement (DSACKs) and Stream Control Transmission Protocol (SCTP) Duplicate Transmission Sequence Numbers (TSNs) to Detect Spurious Retransmissions", RFC 3708, February 2004.

[RFC4015] Ludwig, R. and A. Gurtov, "The Eifel Response Algorithm for TCP", RFC 4015, February 2005.

[RFC5681] Allman, M., Paxson, V., and E. Blanton, "TCP Congestion Control", RFC 5681, September 2009.

[RFC5682] Sarolahti, P., Kojo, M., Yamamoto, K., and M. Hata, "Forward RTO-Recovery (F-RTO): An Algorithm for Detecting Spurious Retransmission Timeouts with TCP", RFC 5682, September 2009.

[draft-kuehlewind-conex-accurate-ecn] Kuehlewind, M. and R. Scheffenegger, "Accurate ECN Feedback in TCP", draft-kuehlewind-conex-accurate-ecn-00 (work in progress), Jun 2011.

Authors' Addresses

Mirja Kuehlewind (editor)
University of Stuttgart
Pfaffenwaldring 47
Stuttgart 70569
Germany

Email: mirja.kuehlewind@ikr.uni-stuttgart.de

Richard Scheffenegger
NetApp, Inc.
Am Euro Platz 2
Vienna,   1120
Austria

Phone: +43 1 3676811 3146
Email: rs@netapp.com